UNIVERSIDAD AUTONOMA DE MADRID

ESCUELA POLITECNICA SUPERIOR

Grado en Ingeniería Informática

TRABAJO FIN DE GRADO

Desarrollo de un sistema de medida de recogida de datos y monitorización de VoIP

Alejandro Nogales García
Tutor: David Muelas Recuenco
Ponente: Jorge E. López de Vergara

Mayo 2015

Resumen

Las mejoras tecnológicas y su progresivo abaratamiento han hecho posible que en los últimos años se haya producido un notable incremento del uso de las tecnologías VoIP. Poco a poco han ido ganando cada vez más terreno en el ámbito de las telecomunicaciones y progresivamente se van sustituyendo las antiguas redes de telefonía por redes de datos IP.

Sin embargo aún queda mucho camino por recorrer para que sea posible una migración completa de la telefonía tradicional a la telefonía VoIP.

Los principales inconvenientes de las redes IP vienen dados por ciertas particularidades de su funcionamiento, las cuales las hacen muy poco apropiadas para la transmisión de datos en tiempo real, como lo es el audio de una conversación telefónica. Es por ello, que VoIP necesita de una monitorización constante de la red para poder gestionar el tráfico y evitar saturaciones en los diferentes enlaces que la componen, para de esta forma poder evitar o reducir lo máximo posible el impacto de la pérdida de paquetes, retraso en las comunicaciones... y así poder ofrecer unos parámetros de calidad similares a los obtenidos bajo una red telefónica.

El grupo HPCN de la UAM ha desarrollado con este objetivo el programa VoIPCallMon, el cual es capaz de filtrar el tráfico de la red para detectar las llamadas VoIP realizadas bajo los protocolos de señalización: SIP, SCCP y Unistim. Con este TFG se pretende ampliar la funcionalidad del sistema VoIPCallMon desarrollando un módulo independiente que permita la detección de las llamadas efectuadas bajo el protocolo de señalización H.323.

A lo largo de la memoria se describirán algunas cuestiones generales de las tecnologías VoIP, con el fin de proporcionar un contexto al desarrollo realizado. Además, se ilustrará una metodología de evaluación de este tipo de herramientas en entornos de virtualización de red, con el fin de demostrar el buen funcionamiento del módulo implementado.

Palabras clave

Voz sobre IP, QoS, QoE, H.323, H.225, Q.931, H.245, RTP, VoIPCallMon, Monitorización, Disección, Wireshark

Summary/abstract

Technological improvements and progressive lowering have caused an increase in the use of VoIP technologies in recent years. Gradually, these technologies have been gaining ground in the field of telecommunications and they are progressively replacing the old telephony networks by IP data networks.

However much remains to be done to make possible a complete migration from traditional telephony to VoIP telephony.

The main drawbacks of IP networks are given by certain peculiarities of its operation, which make them very unsuitable for data transmission in real time, such as the audio of a phone conversation is. It is for this reason that VoIP requires constant monitoring of the network to manage traffic and avoid saturation in the various links that compose it, and in this way to avoid or reduce as much as possible the impact of packet loss, delay communications ... so we can provide quality parameters similar to those obtained on a telephone network.

The HPCN group in the UAM has developed for this purpose the VoIPCallMon program, which is able to filter the network traffic to detect VoIP calls made under several signaling protocols: namely, SIP, SCCP and Unistim. With this TFG we extend the functionality of the system VoIPCallMon by developing a separate module that allows detection of calls made under the H.323 signaling protocol.

Throughout this document, we will describe the general context of VoIP technologies in order to provide a background for the development that was performed and that will be also extensively described. In addition, we will describe a methodology for evaluating such type of tools in network virtualization environments, in order to demonstrate the proper functioning of the implemented module.

Index terms

Voice over IP, QoS, QoE, H.323, H.225, Q.931, H.245, RTP, VoIPCallMon, Monitoring, Dissection, Wireshark

Índice

Índice de figuras

Acrónimos

ASN.1. Abstract Syntax Notation 1.

FCAPS. Fault, Configuration, Accounting, Performance, Security.

HPCN. High Performance Computing Network.

IETF. Internet Engineering Task Force.

ITU. International Telecommunication Union.

IAX. Inter Asterisk Exchange.

IP. Internet Protocol

MCU. Multipoint Control Units.

MGCP. Media Gateway Control Protocol.

MOS. Mean Opinion Score.

MP. Multipoint Controler.

PER. Packet Encoding Rules.

PTSN. Public switched telephone network.

QoS. Quality of Service.

QoE. Quality of Experience.

RTP. Real-time Transport Protocol.

RTCP. RTP Control Protocol.

SCCP. Skinny Call Control Protocol.

SIP. Session Initiation Protocol.

SO. Sistema Operativo.

TCP. Transport Control Protocol.

UDP. User Datagram Protocol.

UNISTIM. Unified Networks IP Stimulus.

VoIP. Voz sobre IP.

Glosario

Ancho de banda. Medida de los datos que pueden enviarse a través de un enlace de red expresada en múltiplos de bit.

Jitter. Medida de la variación de la latencia sobre un determinado periodo de tiempo.

Latencia. Tiempo empleado por un paquete para ir desde su origen a su destino.

MOS. Medida subjetiva de la calidad percibida por un usuario al transmitir o recibir audio o video. Ha sido empleado para medir la calidad de las conversaciones telefónicas durante décadas. El MOS se obtiene a partir de una serie de preguntas estándar realizadas a los participantes de una llamada.

QoS. Calidad de los parámetros de funcionamiento de un cierto servicio. En el caso de llamadas realizadas sobre una red VoIP, la QoS se mide en base a una serie de parámetros cuantificables, como el jitter, la latencia, la cantidad de paquetes perdidos, el ancho de banda…

QoE. Calidad percibida por los usuarios de una red VoIP. A diferencia de QoS QoE mide la experiencia del usuario final de la aplicación. Se trata, por tanto, de una medida subjetiva.

Red IP. Red de comunicaciones que utiliza el protocolo IP, y que se basa en la conmutación de paquetes entre un origen y un destino. Estos paquetes pasan a través de distintos nodos, los cuales se encargan de redirigirlos empleando distintos algoritmos de rutado hasta que alcanzan su destino.

Red PTSN. Redes telefónicas o de conmutación de circuitos. En ellas, para que dos extremos se comuniquen se debe establecer previamente un circuito o canal único desde el origen al destino.

Timestamp. Secuencia de caracteres alfa numéricos que indican la fecha y la hora en la que un determinado evento es registrado por una computadora.

1. Introducción

En los últimos años se ha producido un notable incremento en el uso de tecnologías de Voz sobre IP (VoIP, por sus siglas en inglés). En el término VoIP se engloban todas aquellas tecnologías que están involucradas en la transmisión de voz entre dos o más participantes a través de una red de conmutación de paquetes IP (*Internet Protocol*).

Esta tecnología, comenzó a desarrollarse en 1973, cuando Dani Cohen implementó el protocolo: "Network Time Protocol", que permitía transmitir voz en tiempo real dentro de la red Arpanet, precursora de Internet. Sin embargo, pasó bastante tiempo hasta que esta tecnología estuvo disponible para el gran público. Hacia 1995 comenzaron a surgir estándares en cuanto a la señalización de las llamadas dentro de Internet. En 1996 se desarrolló el protocolo de señalización H.323, y en 1999 se publicó la primera especificación del protocolo SIP. Ambos protocolos, abiertos, son de los más utilizados en la actualidad, junto con otros muchos de carácter cerrado, desarrollados por compañías privadas. A partir de 2004 comienzan a proliferar gran cantidad de aplicaciones, tanto gratuitas como de pago, que permiten la transmisión de audio y vídeo a través de Internet, y que posibilitan la comunicación entre millones de personas a lo largo de toda la geografía mundial con un bajo coste monetario.

Actualmente existen multitud de soluciones VoIP que permiten realizar llamadas, videollamadas y conferencias entre varias personas, que son empleadas a diario para cubrir diferentes necesidades. Para entender la gran difusión de estas tecnologías, podemos señalar las posibilidades que se abre a partir de su uso e implantación. Entre otras aplicaciones, resultan particularmente llamativas las asociadas al ámbito de la educación [1], facilitando la prestación de servicios remotos a usuarios que no ya no requieren acceso físico a las infraestructuras clásicas (por ejemplo, aulas); a diversos servicios sanitarios [2], permitiendo incluso llevar a cabo operaciones guiadas a distancia ante la imposibilidad de disponer de un especialista en un momento preciso; o a ciertas labores de justicia, ya que estas tecnologías se emplean para presentar declaraciones ante los jueces, evitando en muchos casos situaciones traumáticas (por ejemplo, que las víctimas de un crimen se enfrenten al acusado) y permitiendo así organizar los recursos de una forma más eficiente.

Además, el avance de las tecnologías de comunicaciones de los últimos años ha permitido una gran mejora en las instalaciones de redes e infraestructuras de telecomunicaciones, lo que ha repercutido en un acceso a Internet más rápido y barato para los usuarios y ha permitido la difusión de las tecnologías de VoIP [3]. La siguiente imagen, extraída de [4], muestra el número de líneas de fibra óptica instaladas en España desde el año 2007 hasta 2013. En el gráfico se puede apreciar con claridad cómo ha ido en aumento.

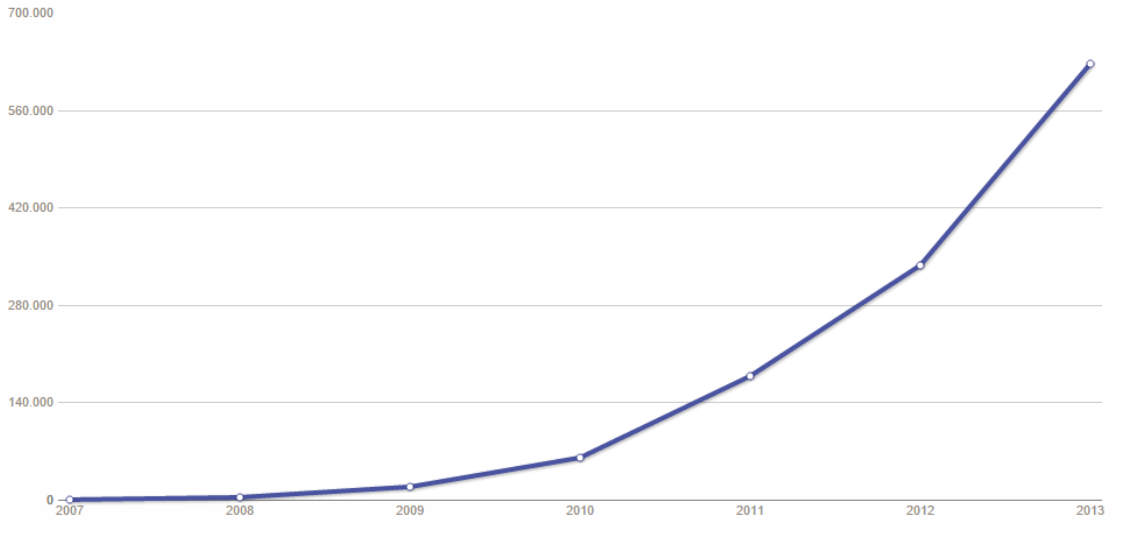


Ilustración 1

Líneas de banda ancha FTTH en España

Gracias a estas mejoras introducidas en la infraestructura de acceso, los proveedores de servicios pueden ofrecer servicios VoIP que compiten con la telefonía tradicional. También las empresas optan por este tipo de tecnología en sus instalaciones, debido a que ofrece una mayor versatilidad que las redes PTSN, pudiendo integrar la transmisión de audio, video y cualquier otro tipo de dato en una sola red.

No obstante, desde el punto de vista técnico, las redes IP presentan ciertas particularidades frente a las redes de telefonía o redes PTSN que hacen que sea necesario establecer un mecanismo de control y seguimiento del tráfico para poder garantizar unos ciertos parámetros de calidad en la transmisión de flujos multimedia. Por ejemplo, IP no ofrece ninguna garantía frente a pérdida de paquetes. Esto significa que, si se sobrepasan las capacidades de transmisión de datos que proporciona la infraestructura de red, se pueden producir pérdidas no recuperables de paquetes que podrían inducir errores en la comunicación. En el caso de transmisión de voz, que es el en el que nos centramos, se puede tolerar una cierta pérdida de paquetes asociada a un descenso de la calidad del audio. No obstante, por encima de un umbral que depende de diversos parámetros, estas pérdidas pueden llegar a imposibilitar la prestación del servicio.

Esta situación hace que sea necesario controlar diversos parámetros de la red con el fin de detectar situaciones problemáticas para el buen funcionamiento de los servicios que dependen de las telecomunicaciones. Es por ello que se han introducido diferentes definiciones y estándares para definir modelos de gestión de red: por ejemplo, FCAPS (*Fault, Configuration, Accounting, Performance, Security*) es el propuesto por ISO en base a los principios incluidos en las recomendaciones de ITU (*International Communications Union*) [5]. El desarrollo de estas actividades permite, por ejemplo, adecuar la velocidad de transmisión dentro de una red al ancho de banda disponible, de forma que no se saturen los elementos de red y se minimice la pérdida de paquetes, mejorando de esta forma la calidad de los servicios prestados.

En particular, los despliegues de VoIP requieren una monitorización particularmente cuidadosa de ciertas medidas de red. El objetivo es que se puedan asegurar unos

parámetros de calidad adecuados para la transmisión de voz, comparables a los obtenidos en redes PTSN. Esta necesidad de elementos adicionales de monitorización que tengan en cuenta medidas interesantes para la transmisión de audio en tiempo real es una de las razones que motivan el proyecto VoIPCallMon, desarrollado por el grupo de Redes y Computación de Altas Prestaciones (HPCN) en la Universidad Autónoma de Madrid.

Este sistema permite la monitorización y análisis de las llamadas de VoIP en redes de alta velocidad con tasas de hasta 10Gb/s. Para detectar y monitorizar las llamadas de VoIP, VoIPCallMon detecta y analiza las conexiones de señalización características de este tipo de aplicaciones para extraer los datos de la transmisión multimedia. Actualmente, VoIPCallMon detecta llamadas efectuadas bajo algunos de los protocolos de señalización más utilizados, en particular SCCP (Skinny Call Control Protocol) o también conocido como Skinny, Unistim, y SIP. VoIPCallMon, por tanto, extrae los parámetros de la conexión de señalización (por ejemplo, direcciones de red de origen y de destino, identificadores de los participantes, duración o motivo de finalización de la llamada,…) y los parámetros más importantes de la conexión multimedia, para posteriormente poder hacer una evaluación de Calidad de Servicio (QoS), así como estimaciones de la Calidad de Experiencia (QoE) ofreciendo estimaciones de medidas subjetivas de calidad tales como MOS.

En este TFG se describe el proceso de desarrollo seguido para implementar un módulo que amplíe la funcionalidad del sistema VoIPCallMon de forma que también se puedan detectar y analizar las llamadas bajo el protocolo H.323. A continuación se exponen los objetivos planteados para este TFG, junto con la planificación seguida para cubrirlos. Posteriormente, se comenta la estructura de este documento, desarrollando su relación con los objetivos expuestos y las tareas indicadas en la planificación.

1.1. Objetivo del TFG

El objetivo principal de este TFG es la implementación de un módulo de detección y análisis de tráfico H.323, correspondiente a la señalización de llamadas de VoIP, que además se integrará con el resto de componentes funcionales del sistema VoIPCallMon [6].

VoIPCallMon, como ya se ha comentado anteriormente, es la herramienta de monitorización, análisis y gestión de despliegues de VoIP desarrollada por el grupo de Redes y Computación de Altas Prestaciones (HPCN) de la Escuela Politécnica Superior (EPS) de la UAM. Esta herramienta ha sido diseñada para alcanzar unas prestaciones de servicio que permitan la monitorización pasiva de llamadas en redes multi-Gb/s con una alta tasa de tráfico.

La funcionalidad de este sistema de monitorización incluye la detección y extracción de información de las llamadas efectuadas, tanto en el plano de señalización como en el plano de transmisión de datos multimedia. Así, proporciona medidas de diferentes parámetros útiles para la gestión de despliegues de VoIP, tales como el número de paquetes que se envían y reciben en cada llamada (lo que puede ayudar a detectar y

evitar situaciones de saturación de la red por problemas de dimensionado o malas configuraciones), problemas derivados de altas latencias (aspecto crítico para la QoS y QoE de la transmisión de voz), o situaciones de alto jitter en las comunicaciones.

En nuestro caso, vamos a desarrollar un módulo que se integre en el sistema VoIPCallMon y amplíe su funcionalidad, añadiendo soporte para el protocolo H.323. Dado que VoIPCallMon ha sido desarrollada para operar a grandes velocidades y con una gran tasa de tráfico de red, el módulo a desarrollar seguirá también estrategias para maximizar la eficiencia de la herramienta. Esta máxima se ha tenido en cuenta desde el inicio del desarrollo de este TFG, de modo que los principios aplicados durante las tareas de análisis y diseño de la solución han sido enfocados para conseguir un elemento software con buen rendimiento que no penalice en gran medida el desempeño general del sistema.

1.2. Fases de realización

El desarrollo de este proyecto se ha planteado en cuatro bloques claramente diferenciados:

1. Investigación y búsqueda de información sobre el estado de la técnica.
2. Análisis, diseño e implementación de la solución.
3. Validación de la herramienta desarrollada.
4. Conclusión del trabajo.

A continuación desarrollamos los principales aspectos de los mismos:

Investigación y búsqueda de información sobre el estado de la técnica. Durante esta primera etapa del desarrollo del proyecto fue necesario recopilar y estudiar la información disponible acerca del protocolo H.323. Este es el primer paso a la hora de desarrollar las herramientas analíticas como la que nos ocupa, ya que es necesario definir una estrategia de detección y análisis de los mensajes, y un modelo de máquina de estados para las conexiones.

También fue necesario comprender el funcionamiento del programa VoIPCallMon. Para ello, se realizaron varios test de ejecución y se analizó el código con el objetivo de comprender su estructura y la organización de los diferentes módulos que lo componen. Dado que el producto software desarrollado a lo largo de la realización de este trabajo debe integrarse en un desarrollo existente, el conocimiento de la estructura interna del programa resultó esencial para instrumentalizar la resolución del problema que motiva el proyecto.

Dadas las características del protocolo a analizar, que serán detalladas posteriormente, surgió la necesidad de utilizar un decodificador para ASN.1. Se evaluaron diversas opciones de licencia libre, entre ellas III ASN.1 Tools [7] y ASN.1 Compiler [8]. Ninguna de las anteriores proporcionó resultados satisfactorios, por lo que fue necesario emplear un generador de decodificadores de ASN.1 comercial, ASN.1 Tools for C [9], desarrollado por OSS Nokalva, que ofrecía una versión de prueba. Adicionalmente, la empresa desarrolladora de este software cedió un código de licencia válido durante el tiempo necesario para realizar el TFG.

Durante esta fase también se decidió montar un sistema de VoIP en una red local. El objetivo de este montaje fue el de disponer de un entorno controlado de pruebas y estudio de la estructura de los mensajes del protocolo H.323. Así, con ayuda del software Wireshark [10] [11], se recolectó un conjunto de trazas de paquetes de red de varias llamadas con diferentes parámetros que han servido para refinar y validar la solución presentada.

Análisis, diseño e implementación de la solución. Esta fase engloba todo el desarrollo del código del proyecto. En este sentido, incluye tanto el desarrollo del módulo para análisis de tráfico de H.323 como la integración de éste módulo en el sistema VoIPCallMon.

El desarrollo se puede dividir en 3 etapas:

- *Librería de decodificación ASN.1*. Se implementó una capa intermedia que permitiese decodificar los datos codificados en ASN.1 en función del tipo de mensaje recibido. Se consideró necesario el desarrollo de esta librería por dos motivos: comodidad, a la hora de decodificar los diferentes tipos mensajes; y, el más importante, independencia, ya que de esta forma se puede aislar la parte del código centrada en el análisis del tráfico de red de la librería empleada para decodificar.

  También se implementó un programa para ejecutar pruebas con los distintos tipos de mensajes necesarios para la detección de la llamada y verificar que la librería era capaz de decodificar todos los tipos correctamente.

- *Implementación de las funciones para el manejo de las diferentes estructuras.* Durante esta etapa del desarrollo se implementaron todas las funciones necesarias para manejar las estructuras de datos donde se almacenan los datos de cada llamada, así como las funciones para manejar las tablas hash y las listas donde se almacenan las llamadas activas.

- *Desarrollo de la máquina de estados*. Comprende el diseño y la implementación de la máquina de estados. Es aquí donde está la lógica principal del módulo. Durante su desarrollo también se realizaron una serie de pruebas para comprobar su correcto funcionamiento.

  Mediante el desarrollo de un programa aparte se probó a simular la recepción de diferentes tipos de mensajes con el objetivo de comprobar que la máquina de estados transitaba correctamente entre los diferentes estados y asegurar que no hubiese problemas con estados muertos.

El hito que marca el final de esta fase es la consolidación del sistema con el módulo que implementa la nueva funcionalidad analítica.

Validación de la herramienta desarrollada. Durante esta fase se montaron varios escenarios con diferentes configuraciones. Se hicieron diferentes ejecuciones del software para, posteriormente recoger los resultados y evaluarlos para determinar el buen funcionamiento de la herramienta en base a los requisitos definidos.

Conclusión del trabajo. Durante esta última fase se valoró el trabajo realizado y se redactó el presente documento en el que se describe el trabajo realizado, los resultados obtenidos y las conclusiones extraídas.

A continuación se muestra una tabla con las principales tareas y subtareas, y el tiempo en horas dedicado a cada una de ellas.

Tarea	**Horas**
Investigación y búsqueda de información sobre el estado de la técnica	
Búsqueda bibliográfica e investigación sobre el protocolo H.323	40
Búsqueda de diferentes alternativas para decodificar ASN.1	30
Análisis, diseño e implementación de la solución.	
Pruebas de las diferentes librerías de decodificación ASN.1	80
Implementación de la librería de decodificación	5
Implementación de los diferentes disectores de mensajes	15
Desarrollo de las pruebas unitarias de los disectores	10
Implementación de la máquina de estados	50
Desarrollo de las pruebas unitarias para la máquina de estados	10
Validación de la herramienta desarrollada.	
Pruebas de integración	10
Pruebas de validación	15
Pruebas empleando software de virtualización de redes	25
Conclusión del trabajo.	
Redacción de la memoria	45
	300

1.3. Estructura de este documento

En este apartado se quiere dar una breve descripción del contenido de este TFG.

- **Capítulo 1.** En este primer capítulo se exponen las motivaciones y objetivos que se desean alcanzar con la realización del TFG. También las diferentes fases de desarrollo del proyecto, así como el tiempo que se ha dedicado a cada fase.

- **Capítulo 2.** Muestra una visión generalizada del estado actual de la técnica, explicando qué se entiende por VoIP, los tipos de redes que se emplean en telecomunicaciones y los diferentes protocolos y estándares más relevantes en este campo. También se describe en que consiste el protocolo H.323 y los elementos que forman una red de estas características.

- **Capítulo 3.** En este capítulo se realizará un análisis de los requisitos que debe cumplir el software que desarrollaremos a lo largo del TFG y se sentarán las bases de diseño para iniciar la implementación.

- **Capítulo 4.** Describe el proceso de implementación seguido para desarrollar el módulo H.323. Se muestran las partes más importantes del sistema, explicando cómo se realiza el filtrado tráfico H.323 y su posterior análisis.

- **Capítulo 5.** Tras la implementación del software, es el momento de evaluar si el software implementado cumple con los requisitos definidos en el Capítulo 3. A lo largo de este capítulo se describen los diferentes escenarios de pruebas y principales soluciones utilizadas. Por último se analizan los resultados obtenidos tras la validación.

- **Capítulo 6.** Para finalizar en este capítulo se expondrán las conclusiones extraídas tras la realización del TFG y se plantearán las posibles líneas de desarrollo para continuar con el trabajo realizado.

2. Estado del arte

2.1. Introducción

A lo largo de esta sección se introducirán algunos de los conceptos necesarios para comprender el trabajo desarrollado. Para ello se comenzará explicando qué se entiende por VoIP para posteriormente enumerar y detallar los diferentes elementos físicos y los principales protocolos de señalización de llamadas que forman parte de un despliegue típico de VoIP. Posteriormente se pasará a explicar en profundidad el protocolo H.323, describiendo todos los elementos que componen un escenario de telefonía en el que se emplea H.323. Por último se presentará la herramienta VoIPCallMon describiendo de forma breve sus partes principales y elementos con la finalidad de contextualizar el módulo desarrollado.

2.2. VoIP

VoIP es un término utilizado para hacer referencia al conjunto de recursos que se emplean para la transmisión de voz a través de redes IP. Las redes IP, a diferencia de las redes comúnmente empleadas para telefonía, las redes de conmutación de circuitos o redes PTSN, basan su rendimiento en la técnica conocida como "best effort" o, en castellano, mejor esfuerzo. Esto es, los paquetes disponen de una dirección de destino y son enviados a la red por los diferentes nodos terminales, y reenviados por los diferentes enlaces que componen la red según su orden de llegada. En el caso de que un enlace reciba más paquetes de los que pueda procesar, éste se satura y no puede reenviar más, por lo que los paquetes que lleguen serán descartados.

Debido a este planteamiento, las redes IP, no ofrecen ninguna garantía al emisor de que el receptor ha recibido el mensaje, ni se garantizan que el paquete recibido no contenga errores, características que no las hacen muy adecuadas para la transmisión de datos en tiempo real, como lo es el audio de una conversación telefónica.

No obstante, es posible garantizar que los paquetes enviados llegan a su destino y que no haya errores en la transmisión del mensaje añadiendo datos extra a la comunicación. Para cumplir este objetivo, se diseñó el protocolo TCP, que trabaja sobre el protocolo IP, y garantiza que los datos enviados lleguen al destinatario sin errores, así como que lo hagan en el orden en que han sido enviados.

A pesar de los inconvenientes que presentan las redes IP para la transmisión de datos en tiempo real, cada vez más, las comunicaciones tienden a realizarse sobre este tipo de redes ya que presentan ciertas ventajas frente a las redes convencionales o PTSN.

Entre sus ventajas destaca que este tipo de redes son mucho más flexibles y fáciles de escalar. También permiten transmitir todo tipo de datos a través de la misma red, lo que facilita la transmisión simultánea de video, audio y datos, frente a las redes PTSN que solo pueden transmitir audio y deben utilizar una red de comunicación de datos aparte para poder transmitir el video.

Los nuevos avances en las infraestructuras de este tipo de redes de comunicaciones han permitido un importante incremento del ancho de banda y de la velocidad de transmisión de datos, así como la disminución de latencias en comunicaciones de larga distancia, lo que hace posible plantear la utilización de este tipo de redes para comunicación en tiempo real. Sin embargo, el principal motivo de esta migración se debe a la gran versatilidad de este tipo de redes, que permiten enviar todo tipo de datos independientemente de su naturaleza, frente a las redes PSTN, pensadas solamente para la transmisión de audio. Esto hace que se adapten mejor a nuevos usos y necesidades, como son la transmisión de audio, video o cualquier tipo de dato, entre dos o más participantes.

Desde hace algunos años se pueden encontrar en el mercado gran variedad de soluciones y herramientas que permiten comunicaciones entre dos o más personas, algunas de ellas gratuitas, entre las más populares destacan por ejemplo Skype y Google Hangouts. Además cada vez son más las empresas de telecomunicaciones y operadores de telefonía que ofertan servicios VoIP, capaces de competir con la telefonía tradicional tanto en calidad como en coste.

Como se ha comentado, existen multitud de aplicaciones disponibles para realizar llamadas VoIP. Estas aplicaciones pueden diferir en muchos aspectos, como el modelo de aplicación distribuida escogido para su funcionamiento, que puede ser tanto de tipo cliente-servidor, como p2p o arquitectura híbrida (mezcla de las dos anteriores). Sin embargo, una característica que los diferencia y que permite establecer una clasificación entre este tipo de programas, es la naturaleza de la licencia empleada para proteger el protocolo de señalización. Así pues, atendiendo a este criterio se puede distinguir entre protocolos abiertos o protocolos propietarios.

Una gran desventaja que presentan las aplicaciones de protocolo propietario o privado, es que requieren que ambos usuarios dispongan de la misma aplicación para realizar una llamada. Son dependientes de la empresa desarrolladora, y en muchos casos sólo ésta puede escribir software que lo implemente, lo que los puede hacer mucho menos versátiles si se trata de una empresa pequeña o con un presupuesto reducido. Un ejemplo de este tipo de protocolo es Skype (tanto la aplicación como el protocolo reciben el mismo nombre), actualmente perteneciente a Microsoft.

Por el contrario los protocolos abiertos, disponen de documentación pública para que cualquier desarrollador pueda implementar el protocolo, esto permite una mayor interoperabilidad al poderse establecer comunicaciones desde cualquier softphone que incorpore el protocolo en cuestión. No obstante no todos los protocolos propietarios tienen el mismo grado de restricción, algunos permiten su uso de forma libre e incluso otros, como Skinny mantenido por Cisco Systems Inc, están perfectamente documentados.

Otro aspecto que podríamos señalar como desventaja de los protocolos propietarios frente a los protocolos abiertos es la seguridad, ya que con los protocolos abiertos al permitir que cualquier persona pueda estudiar sus especificaciones de diseño e implementarlo, es mucho más probable que se subsanen errores de diseño que en los propietarios, donde solo un reducido porcentaje de personas tiene acceso a la especificación del protocolo, en las que depende en gran medida del presupuesto asignado a su mantenimiento.

Podemos decir entonces que uno de los aspectos más importantes a la hora de diferenciar los diferentes sistemas de VoIP, es el protocolo empleado para la señalización de la llamada, es decir, el conjunto de mensajes intercambiados entre los diferentes elementos de la red VoIP para indicar desde el comienzo hasta la finalización de la llamada, incluyendo también los mensajes necesarios para la negociación de capacidades y canales. Dependiendo del protocolo elegido, pueden variar características determinantes, como la calidad de la comunicación, puesto que existen diferencias importantes en la cantidad de información que envían a la red unos u otros para señalizar las llamadas, pudiendo en ocasiones saturarla. Además según el protocolo serán soportadas ciertas operaciones adicionales sobre las llamadas, como el desvío, la puesta en espera, añadir autenticación de los participantes o cifrado a las conversaciones...

Entre los protocolos de señalización más extendidos están SIP o H.323. Ambos presentan diferencias sustanciales [12]. Otros protocolos también con bastante difusión son SCCP, IAX/IAX2 o MGCP.

Para la transmisión de flujos de audio o video es común emplear el protocolo RTP, definido en el RFC3550 [13], normalmente acompañado del protocolo RTCP. RTCP es utilizado para controlar los parámetros de la comunicación RTP, regulando algunos parámetros con el fin de ofrecer una mayor QoS.

A continuación se describen brevemente las propiedades más reseñables de cada protocolo:

- **SIP (Session Initiation Protocol).** Es uno de los más empleados en la actualidad. Se trata de un protocolo bastante reciente en comparación con los aquí presentados, la primera especificación fue publicada en 1999 por la IETF. Se caracteriza por su sintaxis, muy parecida a la empleada por otros protocolos usados en la web, como HTTP o SMTP. En este tipo de protocolos los mensajes son enviados en texto plano, por lo que son muy sencillos de interpretar y facilitan las labores de gestión y control para los administradores de red. Sigue un modelo cliente-servidor, y los mensajes de señalización pueden ser encapsulados bajo los protocolos TCP o UDP.

- **SCCP (Skinny Call Control Protocol).** Originalmente desarrollado por Selsius Corporation y actualmente mantenido por Cisco Systems, Inc. Presenta una arquitectura de tipo cliente-servidor en la que los diferentes clientes o terminales se conectan al Call Manager. El Call Manager es el elemento encargado de realizar la gestión de las llamadas. Como protocolo de transporte para los mensajes se emplea TCP. Destaca su ligereza [14], ya que fue diseñado para ser empleado en dispositivos con limitaciones de memoria o una reducida capacidad de procesamiento.

- **IAX/IAX2 (Inter Asterisk Exchange).** Se trata de un protocolo binario. IAX2 es la evolución del protocolo IAX. IAX2 se define en la recomendación RFC-5456 [14]. El protocolo es un estándar abierto aunque en su mantenimiento también colabora la empresa Digium. Utilizado para la comunicación entre distintas centralitas Asterisk, se trata de un protocolo bastante simple que requiere poco ancho de banda. Asterisk es un proyecto de software libre que

provee funciones similares a una centralita telefónica en una red VoIP. No obstante, dispone de gran multitud de códecs para audio y permite manejar varios streams multimedia en un único flujo, algo que resulta bastante interesante por ejemplo cuando se realizan multiconferencias. Los mensajes de señalización se transportan sobre UDP. El audio se transmite in-band, esto es, en el mismo flujo de comunicación que el empleado por los mensajes de señalización.

- **MGCP (Media Gateway Control Protocol).** Al igual que SIP, también está diseñado siguiendo una arquitectura cliente-servidor en la que los clientes se conectan a un elemento intermediario que es el encargado de negociar la llamada. Definido en la recomendación RFC 3435 [15], no está considerado un estándar. En un escenario MGCP están presentes los sientes elementos: un Media Gatekeeper Controller (MGC), uno o más Media Gateways (MG) y uno o más Signaling Gateways (SG). Así, este protocolo divide el procesado del audio de las operaciones de control de la llamada. De esta forma, el MG se ocupa de gestionar el contenido multimedia, el MGC controla la señalización de la llamada dentro de la red IP y el SG se encarga de la señalización de la llamada bajo la red PSTN.

2.2.1. Arquitectura de una red VoIP

En términos generales, los despliegues de VoIP comparten una arquitectura similar independientemente del protocolo que se utilice para la señalización de la llamada. Habitualmente esta arquitectura incluye los siguientes elementos:

- **Puntos terminales.** Son los puntos finales en la comunicación. Son todos aquellos dispositivos e interfaces de usuario como altavoces, micrófonos, monitores… usados para el envío y recepción de audio. También existen en el mercado teléfonos IP, que son dispositivos similares a un teléfono común solo que disponen de un puerto Ethernet (RJ-45) en lugar de una conexión telefónica tradicional (RJ-11).

Ilustración 2

Teléfono IP (fab. Cisco)

Ilustración 3

RJ-45

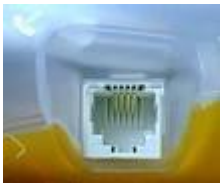


Ilustración 4

RJ-11

- PBX. También conocidos como gatekeepers. Actúan como centralita de la red VoIP. Es donde se deben registrar distintos terminales integrantes de la red, para poder ser localizados. Actúa también como traductor de direcciones y localizador de usuarios.

- **Gateways.** Estos son los enlaces que permiten la interconexión con la red telefónica tradicional PTSN. Hacen posible que las llamadas pasen de una red IP a otra PTSN o viceversa de una forma totalmente transparente para los usuarios.

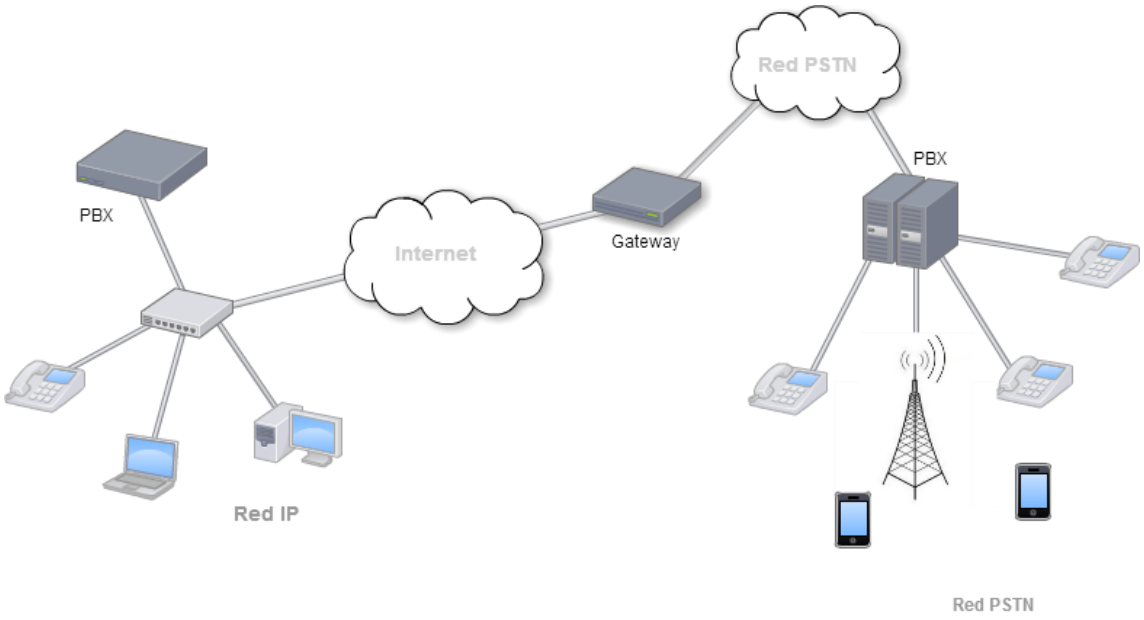


Ilustración 5

Ejemplo de interconexión entre una red IP y una red PSTN

Una vez se han descrito los principales elementos que constituyen una red de VoIP, vamos a considerar el procedimiento habitual mediante el cual se establece una llamada. Este procedimiento no es independiente del protocolo utilizado, aunque casi todos los protocolos comparten una serie de características y fases de negociación similares.

Inicialmente el participante que inicia la comunicación debe iniciar la comunicación con la centralita o PBX utilizando un cierto protocolo de señalización. La centralita se encarga, entre otras funciones, de:

- **Verificar las credenciales de los usuarios**, si el usuario llamante dispone o no de crédito (en el caso de los servicios de pago) o si el usuario puede llamar al número de destino (este puede no estar disponible).

- **Negociar la llamada**, decidir si se trata de una llamada de audio solamente o también incluye video.

- **Negociar el códec** empleado para la compresión del audio y/o el video transmitido.

- **Negociar canales y puertos para los flujos de audio y/o video.**

Una vez que se han decidido todos los parámetros anteriores, los participantes empiezan a enviar flujos de audio y/o video mediante un protocolo de red adecuado (generalmente RTP). La llamada concluye cuando alguno de los participantes envia un mensaje de finalización.

2.3. Protocolo H.323

H.323 es el primer protocolo de señalización de VoIP que fue considerado un estándar internacional. Es una evolución del protocolo SS7 (Sistema de señalización por canal común nº7). La primera versión de este protocolo fue desarrollada por la ITU (International Telecommunication Union) en 1996. La última revisión fue aprobada en diciembre del año 2009 [16].

H.323 es un protocolo ideado para permitir el intercambio de datos multimedia sobre redes de paquetes. Entre sus principales características destacan [17]:

- **Independencia.** Es independiente de la red, ya que puede trabajar sobre cualquier red IP, tanto sobre TCP como UDP; y de plataformas y aplicaciones.

- **Interoperabilidad.** Entre los diferentes elementos que componen la red, encargándose el protocolo de la negociación de capacidades para dar cabida al mayor número de dispositivos posible.

- **Gestión de red**. H.323 permite controlar el ancho de banda dedicado a los flujos de audio, video o datos con la finalidad de evitar congestiones en la red. Además dispone de medios para monitorizar parámetros de QoS. Otra característica es la posibilidad de gestionar llamadas a crédito.

- **Seguridad.** El protocolo dispone de procedimientos para permitir el cifrado de datos y la autenticación de participantes.

- **Disponibilidad.** H.323 permite incluir elementos redundantes para que, en caso de caída o fallo de uno de sus elementos, el despliegue de VoIP pueda seguir en funcionamiento. Además, se trata de un protocolo robusto ante pequeños errores de comunicación.

- **Soporte para multiconferencias y transmisión en multicast.**

- **Establecimiento rápido de la llamada.** Aunque normalmente se intercambian varios mensajes antes de iniciar una llamada, H.323 dispone de un procedimiento, llamado “fastconnect” o “faststart” mediante el cual solo es necesario el intercambio de dos mensajes.

- **Servicios suplementarios**. H.323 ofrece una serie de servicios suplementarios como:

    - Transferencia de llamada.
    - Desvío de llamada.
    - Llamada en espera.
    - Identificación del número llamante.
    - Priorización de llamadas.
    - Control de destino de la llamada.

Una característica esencial de H.323 es que, además de ser un protocolo binario, usa un sistema de codificación abstracto para los mensajes de señalización y control de llamada denominado ASN.1 (Abstract Syntax Notation) junto con un conjunto de reglas denominadas PER (Packet Encoding Rules). Estas son utilizadas para convertir los datos en un flujo binario para su envío por la red. Esta particularidad hace necesario decodificar previamente los mensajes antes de poder analizarlos.

La estructura de estos elementos funcionales se establece y estandariza en diversos documentos. La sintaxis ASN.1 está definida en las especificaciones X.680-683 [18] de la ITU. Las reglas de codificación PER se definen en la especificación X.691 [19]. El documento en el que se describe H.323 es una especificación "paraguas" creada con el objetivo de definir la forma en que operan entre sí otros protocolos definidos por la ITU, como RAS, H.225 o H.245. Este documento, por lo tanto, engloba diferentes protocolos, de modo que para entender el protocolo H.323 es necesario estudiar también las especificaciones de H.225 [20] y H.245 [21].

En ese sentido, cada uno de estos protocolos se centra en aspectos diferentes. Mientras que H.323 especifica los componentes del sistema, H.225 describe tres protocolos de señalización diferentes, RAS, Q.931 y el protocolo conocido como Anexo G/H.225. H.245 es empleado para: abrir y cerrar los canales lógicos mediante los cuales se canalizan y transmiten los flujos de audio, video o datos; para el negociado e intercambio de capacidades; o para definir los roles en la comunicación.

En la siguiente imagen se puede ver la pila de protocolos que componen H.323.

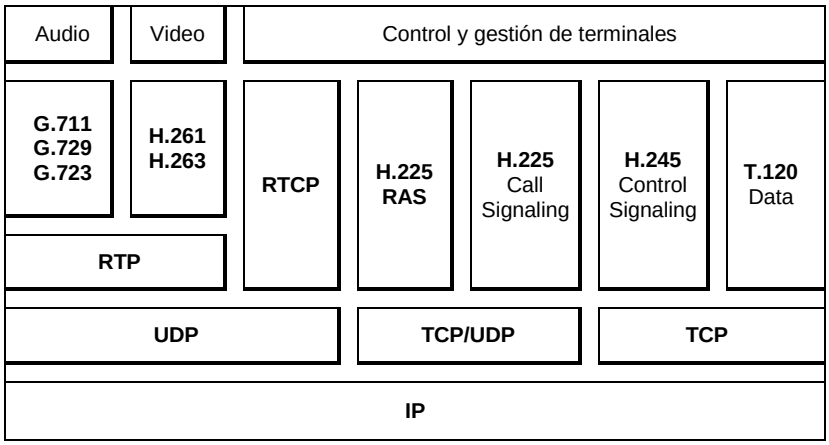


Ilustración 6

Pila de protocolos H.323

2.3.1. Arquitectura general

La arquitectura general de un sistema H.323 [22] consta de los siguientes cuatro elementos:

- **Terminales.** H.323 no impone restricciones sobre las interfaces de red. Sin embargo, con la finalidad de ofrecer interoperabilidad entre diferentes dispositivos, todo terminal debe permitir el intercambio bidireccional de datos de voz y audio. Como requerimientos opcionales se encuentra el intercambio de datos y video. Para facilitar la interoperabilidad, H.323 también ofrece posibilidad de emplear variedad de códecs de video y audio.

- **Gateways.** Ofrecen interoperabilidad entre diferentes tipos de red, incluyendo las redes telefónicas de conmutación de circuitos. El Gateway es el encargado de traducir y guardar la correspondencia de las direcciones y los mensajes de señalización y control entre diferentes tipos de redes.

- **Gatekeepers.** Los gatekeepers son un elemento opcional en los sistemas H.323. Se encargan de realizar varias funciones. pudiendo ser empleados como mecanismo de control para la gestión del tráfico de red. El gatekeeper puede decidir si aceptar o rechazar una llamada en función de la carga de la red, así como limitar el ancho de banda asignado a una determinada llamada. También son utilizados para traducir direcciones, convirtiendo direcciones de teléfono y alias a direcciones de red.

- **MCU (Multipoint Control Units).** También son elementos opcionales. Un MCU se compone de un MC (Multipoint Controler), encargado de gestionar la señalización entre los diferentes puntos terminales y manejar las direcciones de cada flujo de datos. Y ninguno o varios MPs (Multipoint Processors) que se encargan de procesar y mezclar los diferentes flujos de audio y/o video en tiempo real. Son utilizados para dar soporte y robustez a las multiconferencias o llamadas multipunto (conversaciones entre tres o más participantes). Podemos distinguir tres tipos:

    - **Centralizada.** Utiliza un MCU para distribuir los flujos de audio y/o vídeo a los diferentes terminales.

    - **Descentralizada.** No es necesario emplear un MCU: cada terminal envía su flujo de video y/o audio en multicast a todos los participantes.

    - **Híbrida.** Combina los dos anteriores. Para soportar este modo es necesario contar con un MCU con al menos un MP.

2.3.2. Protocolos H.323

A continuación se describen los protocolos H.225 y H.245, indicando la estructura de sus mensajes.

Protocolo H.225

La especificación de H.225 [20] define los mensajes RAS y Q.931 que se emplean tanto para la señalización de la llamada como para la comunicación entre terminales y gatekeepers.

Los mensajes Q.931 se utilizan para la señalización de la llamada, esto es, son los transmitidos entre los extremos terminales de la comunicación para iniciar y finalizar la llamada, y los empleados para controlar la llamada durante su transcurso. Entre los mensajes Q.931 más relevantes destacan los siguientes:

- Setup
- Call Proceeding
- Alerting
- Information
- Release Complete
- Facility
- Progress
- Status
- Status Inquiry
- Setup Acknowledge
- Notify
- Connect

Por otra parte, el protocolo RAS *(Registration, Admission and Status*), es utilizado para la comunicación entre terminales y gatekeepers. Principalmente cumple las siguientes funciones:

- Permite a los terminales localizar el/los diferentes gatekeepers disponibles.

- Registro del terminal en el gatekeeper. Los terminales de una red H.323 deben registrarse con un gatekeeper si éste está disponible, de esta forma, otros terminales puedan localizarlos solicitando la dirección al gatekeeper.

- Informar de la admisión del terminal al gatekeeper. El gatekeeper puede decidir si una llamada se puede realizar o no, por diversos motivos.

- Des-registrar el terminal del gatekeeper.

  Solicitar peticiones de estado. De esta forma el gatekeeper puede conocer si un terminal está activo o inactivo.

- Solicitar cambios del ancho de banda asignado a la comunicación.

Protocolo H.245

El protocolo H.245 se encarga de 3 funciones durante la llamada:

- **Apertura de los canales lógicos.** Un canal lógico consiste en una dirección IP y un número de puerto al que enviar el tráfico RTP. Estos canales son unidireccionales y se emplean para transmitir el audio de una conversación, por lo que para una misma llamada es necesario establecer dos canales

- **Negociación de capacidades.** Los terminales H.323 pueden decidir entre una gran variedad de códecs para audio y video con la finalidad de poder soportar un mayor número de dispositivos. Además de los códecs, también se negocian otros parámetros que afectan a la transmisión de los datos multimedia.

- **Determinación del maestro y esclavo de la comunicación.** Aunque en conversaciones entre dos participantes este mecanismo no es necesario, se hace imprescindible cuando interviene un número mayor de participantes.

Los mensajes H.245 pueden ser de cuatro tipos diferentes:

- **Request.** Este tipo de mensajes son empleados cuando un terminal desea realizar al otro algún tipo de petición. Los siguientes ejemplos son mensajes de este tipo:

    - masterSlaveDetermination, empleado para determinar quién tiene el papel de maestro o esclavo en una comunicación.
    - terminalCapabilitySet, usado para indicar determinados parámetros en una comunicación.
    - openLogicalChannel, empleados para indicar que se desea abrir un canal lógico.

- **Response.** Son los utilizados para dar respuesta a los solicitudes de los mensajes tipo request. En respuesta a los anteriores, por ejemplo, se tienen:

    - masterSlaveDeterminationAck.
    - terminalCapabilitySetAck.
    - openLogicalChannelAck.

- **Command.** Empleados para enviar órdenes a otro terminal. Algunos ejemplos son los siguientes:

    - sendTerminalCapabilitySet.
    - Flow Control.
    - Encryption.
    - End session.

- **Indication.** Son empleados para enviar información cuando no se requiere una respuesta ni ninguna acción por parte del destinatario. Ejemplo de este tipo de mensajes son:

    - UserInput
    - Jitter Indication.
    - Function Not Understood.

En la siguiente imagen se marcan los distintos protocolos involucrados en una llamada H.323 y qué mensajes son intercambiados entre los elementos que componen la red durante el transcurso de la comunicación.

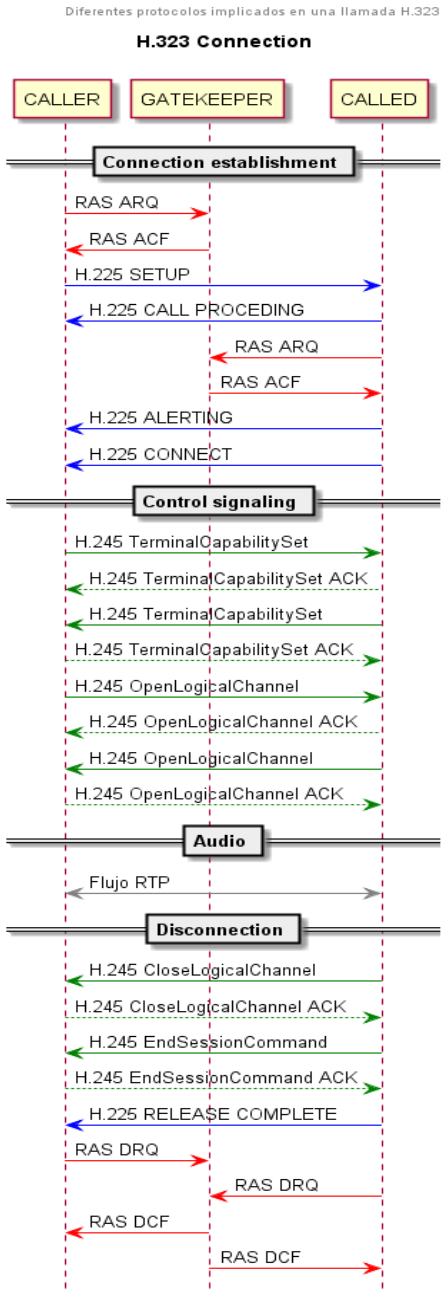


Ilustración 7

Ejemplo de conexión H.323

Como se puede observar, en primer lugar el terminal que inicia la llamada se registra contra el gatekeeper mediante el protocolo RAS. Una vez se ha registrado envía el mensaje "SETUP" para indicar que quiere iniciar la comunicación. El terminal llamado envía un "CALL PROCEEDING" y se registra mediante RAS con el gatekeeper. Una vez que el gatekeeper ha aceptado al otro terminal, este envía los mensajes "ALERTING" y "CONNECT" para indicar que desea aceptar la llamada.

Una vez terminado el procedimiento de señalización de la llamada, entra en juego el protocolo de control H.245. Mediante los mensajes de tipo TerminalCapabilitySet, se negocian los códecs y características de los flujos multimedia, mediante los del tipo OpenLogicalChannel se establecen los canales necesarios para la transmisión y recepción del audio.

Cuando ambos participantes conocen la dirección para emitir el flujo de audio, se inicia el protocolo RTP, protocolo diseñado para la transmisión de datos multimedia, como audio o video.

En el momento en que alguno de los participantes desea terminar la llamada, éste envía un mensaje de tipo "RELEASE" o "RELEASE COMPLETE" para indicarlo. En ese momento, se corta la comunicación y ambos participantes se lo indican al gatekeeper.

El procedimiento descrito es el seguido habitualmente para el establecer una llamada. No obstante, H.323 dispone de un mecanismo mediante el cual se puede iniciar la comunicación enviando solamente dos mensajes. Para ello en el mensaje SETUP se tunelizan varios elementos faststart que contienen información sobre los códecs que se van a utilizar y una serie de elementos OpenLogicalChannel para decidir el canal lógico por el que se enviará el audio de la conversación. Cuando el otro participante responde con el mensaje CONNECT se envían tunelizados otra serie de elementos que contienen los mensajes de respuesta a las peticiones del SETUP y junto con uno o varios elementos OpenLogicalChannel para poder decidir el canal usado para el audio en la otra dirección.

En la siguiente captura se muestra un pequeño extracto de un mensaje SETUP capturado mediante Wireshark, en el que se puede ver cómo se encapsulan los diferentes elementos faststart.

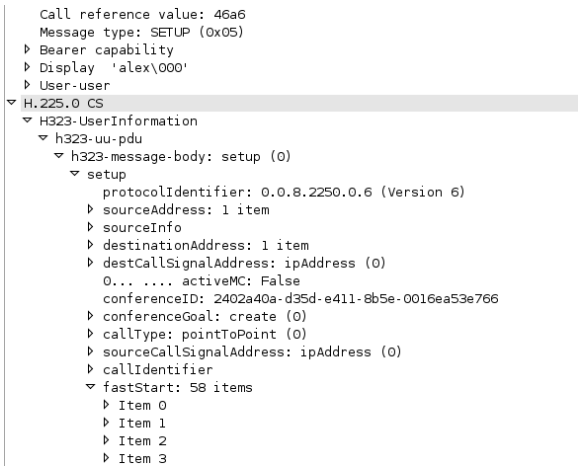


Ilustración 8

Ejemplo de mensaje SETUP

2.4. VoIPCallMon

La extensión desarrollada en este TFG se integra en un proyecto llamado VoIPCallMon. En esta sección se explica en qué consiste este proyecto, y se describe brevemente su funcionamiento con la finalidad de dar una idea clara y concisa de qué lugar ocupa este nuevo módulo dentro del proyecto.

El proyecto VoIPCallMon surge motivado por la notable expansión de los despliegues de VoIP. Las ventajas de esta tecnología frente a las redes telefónicas tradicionales (PSTN) hacen que cada vez más empresas opten por este tipo de comunicaciones. En este escenario de cambio, VoIPCallMon tiene como objetivo permitir la monitorización del tráfico VoIP ya que, debido a las peculiaridades de esta tecnología, resulta necesario registrar ciertos parámetros de la comunicación (tasa de errores, latencias, ancho de banda…) para poder ofrecer la mejor QoS.

Además, si se desea implantar esta tecnología masivamente, también será necesario que los proveedores de servicios puedan monitorizar este tipo de llamadas para poder obtener datos que permitan acometer las tareas de las áreas funcionales de gestión de red (FCAPS). Algunos parámetros interesantes son la identificación del llamante y el llamado, el momento en que se ha efectuado la llamada, o la duración de la llamada. Estos datos son necesarios, por ejemplo, para cumplir con leyes que obligan a mantener a las empresas de telecomunicaciones registros de las llamadas efectuadas, así como también pueden ser empleados para muchos otros usos, como la tarificación de las llamadas u ofertar diferentes tipos de servicios como bloqueo de llamadas, identificación del llamante...

Hay que destacar que todo este proyecto está orientado a ejecutarse bajo un hardware de uso común, es decir, de coste reducido y disponible para cualquier persona, y consigue monitorizar tráfico en redes de alto rendimiento con tasas de hasta 10Gb/s.

A continuación se describe cómo está dividido el programa, indicando a grandes rasgos las funciones llevadas a cabo por cada componente. VoIPCallMon se estructura en cuatro módulos:

- **Módulo de captura de tráfico.** Se encarga de capturar y filtrar el tráfico de red para pasar al siguiente módulo los paquetes que pertenezcan a cualquiera de los protocolos relacionados con VoIP soportados (SIP, SCCP, Unistim o RTP). En [1] se explica en detalle el método seguido para la captura del tráfico en redes de alta velocidad empleando hardware convencional.

- **Módulo encargado de realizar el seguimiento del tráfico VoIP.** Este módulo se encarga de, una vez recibido el paquete, analizarlo y comprobar que efectivamente se corresponde a un mensaje de alguno de los protocolos antes mencionados. Si es así, primero comprueba si se trata de una llamada en curso o es una nueva llamada. Posteriormente, se analiza el mensaje con la finalidad de buscar información sobre el emisor y receptor, la dirección IP y los puertos empleados para transmitir el flujo RTP.

- **Módulo encargado de la generación de estadísticas y de la construcción de los flujos de audio partir del tráfico capturado.** Se encarga de extraer toda la

información y generar las estadísticas de una llamada, una vez ésta ya ha finalizado. Entre los datos recogidos están los indicados en la directiva de la unión europea 2006/24/EC. También recoge algunos parámetros para poder monitorizar la QoS (Quality of Service).

- **Módulo de retención de llamadas.** Este módulo se encarga de volcar periódicamente los datos obtenidos a una base de datos para que así los posibles usuarios de la aplicación puedan manejar y observar cómodamente los datos mediante un front-end web.

2.5. Conclusiones

Como se ha visto a lo largo de este capítulo, VoIP es una tecnología establecida y con un futuro muy prometedor. Sin embargo, dadas las características de las redes IP, es necesario establecer una serie de procedimientos desde el nivel de aplicación que permitan garantizar una QoS y QoE comparable a las obtenidas bajo redes PSTN.

Este capítulo también se ha analizado las características generales de los despliegues de VoIP. Hemos visto como en el contexto general de los despliegues de VoIP, conviven muchos protocolos de señalización de llamadas. Entre los más difundidos se encuentran SIP y H.323, ya que aunque H.323 es más antiguo, está más desarrollado que SIP y dispone de una mayor cantidad de operaciones adicionales que promueven que sigue siendo ampliamente empleado en instalaciones comerciales operativas.

También hemos repasado la estructura de VoIPCallMon, la herramienta de monitorización y análisis en la que se integra el módulo desarrollado en este TFG. VoIPCallMon proporciona diversas medidas de red que permiten detectar problemas en la red con el fin de proporcionar mejores QoS y QoE.

3. Análisis del problema

3.1. Introducción

En este capítulo se van a exponer las especificaciones y requisitos que debe cumplir el sistema que se desarrolla para este TFG.

Una de las fases esenciales durante el desarrollo de todo proyecto software es la etapa de análisis de requisitos. En esta fase, se debe observar y analizar la funcionalidad que deberá cumplir el programa con el objetivo de poder determinar qué tareas deberán realizarse para terminar con éxito el proyecto. Es por ello que es necesario prestarle una gran atención, ya que, dependiendo de cómo sea su desarrollo será más probable detectar y prevenir errores por incompatibilidades, dependencias... u otros riesgos asociados al desarrollo del proyecto.

En nuestro caso, al no tratarse de un nuevo proyecto, sino, que lo que se va a desarrollar es una extensión que amplíe la funcionalidad de un sistema existente, antes de realizar el análisis de requisitos era necesario recopilar cierta información previa. Entre la información consultada a resultado de gran utilidad el documento [1]. En él se describe con gran detalle la implementación y la organización de los diferentes módulos del programa VoIPCallMon. Entender cómo funciona el software VoIPCallMon era esencial para poder integrar con éxito la extensión desarrollada.

También ha sido necesario leer y recopilar información sobre la arquitectura general de los sistemas H.323 [2], para poder conocer que elementos intervienen en una llamada. Además destacar el uso de las especificaciones de la ITU (H.323, H.225 y H.245) para saber que mensajes deben enviarse y recibirse para mantener una conversación.

En los sucesivos apartados se plantean los requisitos que deberá cumplir la solución desarrollada. En primer lugar se hará una breve descripción empleando un lenguaje coloquial para entender mejor el requerimiento a evaluar y posteriormente se formulará con un lenguaje más técnico con la finalidad de no dejar lugar a posibles errores de interpretación.

3.2. Análisis de requisitos

Atendiendo a la funcionalidad que se aborda podemos distinguir dos tipos de requerimientos: requisitos funcionales y requisitos no funcionales.

Los requisitos funcionales son aquellos que describen el funcionamiento que debe ofrecer el sistema que se va a desarrollar. Como requisitos funcionales solamente son considerados aquellos referidos a las operaciones que el software realiza.

Por otra parte, los requisitos no funcionales pueden ser de una gran variedad de tipos, como por ejemplo de fiabilidad, de escalado, de velocidad, de usabilidad... Describen también la funcionalidad del sistema, pero atendiendo a cómo deben realizarse determinadas operaciones o tareas.

3.2.1. Requisitos funcionales

Las acciones que deseamos que lleve a cabo la extensión a desarrollar son:

- La extensión implementada debe ser capaz de permitir el análisis y seguimiento de las llamadas realizadas en una red IP empleando el protocolo H.323. Para ello deberá ser capaz de filtrar los paquetes de la red detectando aquellos que pertenecen a los protocolos que queremos analizar.
- Deberá registrar cada llamada que se efectúe en la red. Almacenando las direcciones IP de ambos participantes de la llamada. Así como sus alias o identificadores empleados dentro de la red H.323. El programa debe ser capaz de distinguir quien es el que efectúa la llamada y quién es quien la recibe. También, deberá registrar la hora de inicio y de finalización de cada llamada. Y, en caso de ser posible, el motivo de la finalización de la llamada, si ha finalizado correctamente o, por el contrario, ha ocurrido algún error que ha obligado a interrumpir la llamada.
- El sistema debe ser capaz de extraer los flujos de audio de la llamada, de ambos sentidos para poder almacenarlo y posteriormente reproducirlo. Para ello será necesario también capturar los códecs de audio empleados en cada llamada para poder determinar cómo se debe reproducir.
- Finalmente el sistema debe ser capaz de almacenar de forma persistente toda la información recogida con la finalidad de que pueda visualizarse de forma cómoda y generar estadísticas de uso de la red mediante un front-end vía web.

Hay que añadir también que para poder analizar los mensajes del protocolo H.323 también será necesario implementar o utilizar una librería de decodificación para la notación ASN.1.

A continuación se muestran de una forma mucho más detallada los requisitos funcionales que se acaban de exponer.

RF1: La extensión se integrará dentro de IPCallMon.

RF2: Debe ser capaz de capturar el tráfico de la red y filtrar los paquetes pertenecientes a los protocolos H.225, H.245 y RTP.

RF3: Se deberán analizar los mensajes H.225. Se mirará en la cabecera Q.931 el tipo de mensaje y según éste se extraerá la siguiente información:

> **RF3.1:** Del mensaje SETUP se obtendrá el identificador de llamada, las direcciones IP tanto del emisor como del receptor del mensaje. Además se debe comprobar si la llamada que se intenta establecer se hace bajo el método normal o mediante el método “fastconnect” y si el protocolo H.245 va tunelizado o en un canal aparte.
>
> **RF3.2:** Call Proceeding. Se extraerá el identificador de llamada y se comprobará si el protocolo H.245 va tunelizado o en un canal aparte.
>
> **RF3.3:** Alerting. Se extraerá el identificador de llamada y se comprobará si el protocolo H.245 va tunelizado o en un canal aparte.

RF3.4: Connect. Se extraerá el identificador de llamada y se comprobará si el protocolo H.245 va tunelizado o en un canal aparte.

RF3.5: Facility. Se extraerá el identificador de llamada y se comprobará si el protocolo H.245 va tunelizado o en un canal aparte.

RF3.6: Connection released. Se extraerá el identificador de llamada y se comprobará si el protocolo H.245 va tunelizado o en un canal aparte. En este tipo de mensajes también será necesario detectar el código que define el estado de terminación de la llamada.

RF4: Se deberán analizar los mensajes H.245

RF4.1: Serán analizados los mensajes de tipo openLocagicalChannelAck, que son los que interesan para nuestro propósito. Se deberán obtener los puertos y la dirección empleada para establecer la comunicación RTP. Será necesario capturar al menos dos mensajes de este tipo para obtener los dos sentidos del audio transmitido.

RF4.2: También se analizarán los mensajes terminalCapabilitySet terminalCapabilitySetAck. De ellos se extraerá la información necesaria para conocer el códec empleado para el audio de la conversación.

RF5: Analizar los mensajes pertenecientes al protocolo RTP.

RF6: Será necesario implementar una librería que permita decodificar los mensajes codificados bajo notación ASN.1.

3.2.2. Requisitos no funcionales

Como requisitos no funcionales se engloban todas aquellas características y funcionalidades que el sistema debe cumplir pero que no son parte fundamental del mismo, como son por ejemplo la robustez frente a errores, un buen rendimiento, la portabilidad... es decir, aquellos que afectan a la forma de realizar las funciones.

Debido a la amplitud de este tipo de requisitos, vamos a destacar los requisitos no funcionales más importantes:

RNF1: Rendimiento. Nuestro objetivo es que el módulo desarrollado no penalice el rendimiento general de VoIPCallMon.

RNF2: Estabilidad. Necesitamos un sistema estable, ya que se va emplear como monitorización de red.

RNF3: Modularidad, el código del disector debe ser diseñado de forma modular para, en un futuro, poder sustituir la librería privativa encargada de decodificar los mensajes codificados en ASN.1 por otra opción con licencia libre.

RNF5: Simplicidad.

Otros requisitos no funcionales que también se tienen en cuenta son:

RNF6: Requisitos organizativos. Documentación del código. Seguimiento del proyecto.

RNF7: Portabilidad. Debe poder ejecutarse en cualquier sistema operativo basado en la distribución de Linux, Debian.

Otros requisitos que podemos considerar son los requerimientos de hardware, ya que la aplicación debe poder ejecutarse en hardware de uso común, por lo que hemos tomado como referencia los ordenadores disponibles en los laboratorios de la EPS.

3.3. Conclusiones

A lo largo de este capítulo se han mostrado ya de una forma más técnica, los diferentes requerimientos que debe cumplir la extensión desarrollada.

El haber realizado este desglose de funcionalidades presenta grandes ventajas a la hora de abordar el diseño y la implementación de la solución, ya que permite disponer de una visión más amplia a la hora de abordar problema, minimizando la aparición de riesgos y maximizando las posibilidades de éxito del proyecto.

En el resto del documento se verá como se ha ido abordando el desarrollo de la solución, se expondrán a fondo los problemas que han surgido y que soluciones se han adoptado para resolverlos.

4. Desarrollo de la solución

4.1. Introducción

En el capítulo anterior se realizó un análisis del problema para poder establecer y redactar los requerimientos de la solución final. A lo largo de todo este capítulo se muestra cómo se ha llegado a esa solución, indicando los procesos y las decisiones más relevantes tomadas durante el proceso de desarrollo.

Comenzaremos proporcionando un vistazo general al diseño de alto nivel de la aplicación; esto es, qué estructuras de datos se utilizan y cómo se organizan para poder manejar la información referente a las diferentes llamadas que se efectúan dentro de la red, explicando adicionalmente cómo se integra la extensión desarrollada dentro del sistema VoIPCallMon.

Posteriormente se explica en detalle cómo se realiza el filtrado del tráfico necesario, y qué mensajes son los involucrados en la llamada H.323, para luego describir cómo es analizado este tráfico.

Por último se comentan las dificultades y particularidades más significativas encontradas durante el proceso de desarrollo, y se extraen una serie de conclusiones que resumen los contenidos de este capítulo.

4.2. Diseño general de alto nivel

El diseño a alto nivel del módulo ha seguido un planteamiento muy similar al seguido en otros módulos componentes del sistema VoIPCallMon, como el de SCCP o SIP. La principal motivación para seguir este diseño es que ya se ha utilizado previamente con éxito en otros módulos, demostrando ofrecer un buen rendimiento a la hora de manejar grandes cantidades de tráfico de datos con miles de llamadas activas simultáneamente. Como ventaja añadida, hay que añadir que al tratarse de un diseño similar se facilita en gran medida la integración con el sistema VoIPCallMon y se mejora las condiciones para las labores de mantenimiento de la herramienta.

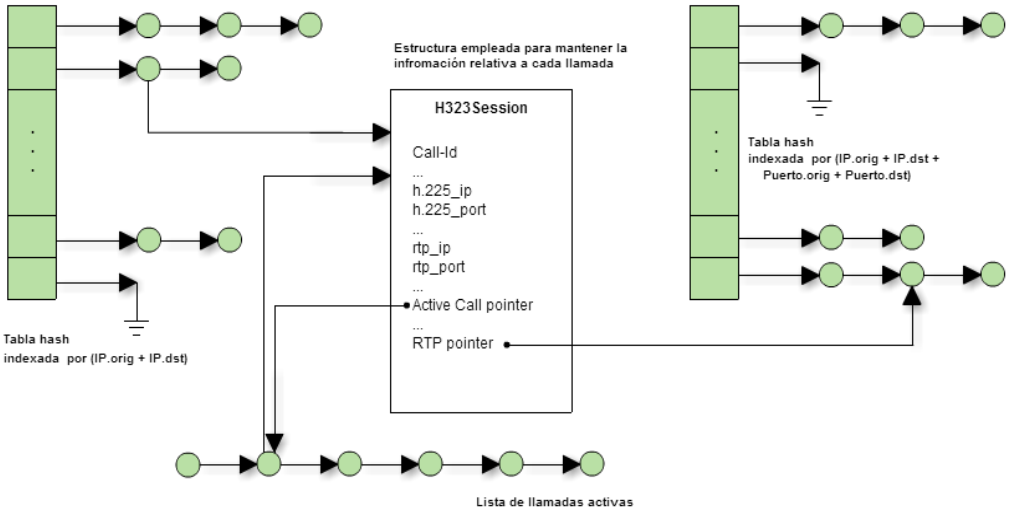


Ilustración 9

Diseño seguido para la implementación del módulo H.323

En la imagen se puede ver el diseño de alto nivel seguido para el módulo. Comentamos a continuación los principales elementos de este diseño:

- Una sesión de tipo H323Session, en la que se almacena la información relativa a cada llamada, manteniendo su estado actual. Este estado indica cómo se realizan las transiciones en la máquina de estados, la cual controla el estado interno de cada llamada y permite conocer el último mensaje recibido. La sesión también se emplea para almacenar los puertos y direcciones IP empleados por H.225, H.245, en el caso de que no valla tunelizado, y RTP. Además contiene un puntero a la lista de llamadas activas, empleado para posicionarla al final cuando se recibe un mensaje (de modo que permanece ordenada en todo momento), y un puntero a la tabla hash de RTP. Más adelante de describe con más detalle los distintos campos de la estructura empleada para almacenar la información de la sesión.

- Una tabla hash que contiene las llamadas que están siendo cursadas. Para indexar las entradas en la tabla se emplea como función hash la suma de las direcciones IP origen y destino de los paquetes H.225. En caso de colisión, se emplea una lista enlazada para añadir las entradas.

- Otra tabla hash con listas enlazadas para el tráfico RTP. Indexada por la suma de IP de origen, IP de destino, puerto de origen y puerto de destino de los paquetes RTP capturados.

- Una lista de llamadas activas. Se trata de una lista ordenada según el instante de llegada del último mensaje recibido. Para mantener este orden, cuando se recibe un nuevo paquete perteneciente a una llamada, ésta se desplaza al final de las lista. La inclusión de esta lista facilita en gran medida las operaciones de búsqueda sobre las llamadas para proceder a la expiración por tiempo.

Descripción del funcionamiento del módulo

El módulo entra en funcionamiento cuando llega un paquete nuevo perteneciente a los protocolos H.225 o H.245. Una vez que hemos detectado que el paquete corresponde a uno de estos protocolos, se calcula su índice sumando las direcciones IP, y se busca si ya existe una entrada en la tabla hash de llamadas.

Si se trata de tráfico H.225 y ya existe una entrada en la tabla, se obtiene la sesión almacenada y se actualiza el estado de la llamada en función del mensaje contenido en el paquete capturado. En el caso de que no exista la entrada, se crea una nueva, se inicializa y se introduce en la tabla en la posición correspondiente al índice calculado.

En el caso del tráfico H.245, miramos si la sesión ya está en la tabla, y en caso de estar la procesamos, si no hay ninguna entrada, descartamos el paquete.

Una vez que disponemos de la información RTP, es decir, los puertos y direcciones IP empleados para transmitir los flujos de audio, pasamos esta información al módulo RTP rellenando una sesión de tipo RTPSession e introduciéndola en la tabla hash de RTP.

Las estructuras empleadas para el manejo de la sesión y para almacenar de forma auxiliar la información relativa a cada llamada están definidas en el fichero "H323.h". A continuación de muestran las más importantes y se señalan los campos más significativos indicando su función.

H323Session

```
typedef struct h323Session {
      uint8_t *payload_ptr;
      uint32_t payload_length;
      unsigned char callId_value[16];
      unsigned short callId_length;
      uint8_t h245Tunnelling;
      CallInfoH323 caller, callee;
      node_l *active_node;
      uint8_t isFaststart;
      Address faststart[H323_TAM_FASTSTART];
      int faststart_choosen;
      enum state rtpCallerInfoState;
      enum state rtpCalledInfoState;
      Address rtpCallerInfo;
      Address rtpCalledInfo;
      int rtpInfo;
      int state;
} H323Session;
```

- **payload_ptr:** puntero a los datos
- **payload_length:** tamaño de los datos apuntados por payload_ptr
- **callId_value:** identificador único de cada llamada, protocolo H.225. Puede ocupar hasta 16 bytes.
- **callId_length:** Necesitamos almacenar la longitud del call-ID porque es variable.

- **active_node:** Puntero a la lista de llamdas activas que contiene la sesión.
- **H245Tunnelling:** Flag que indica si el protocolo H.245 va tunelizado en H.225 o va en un canal aparte.
- **IsFaststart:** Flag para indicar si una llamada sigue el procedimiento habitual para establecerse o emplea el método fastconnect.
- **caller** y **callee:** Almacenan la información relativa a los participantes de la llamada. De tipo CallInfoH323. En el siguiente se describe con más detenimiento.

CallInfoH323

```
typedef struct callInfoH323 {
      uint32_t h225_ip;
      uint16_t h225_port;
      uint32_t h245_ip;
      uint16_t h245_port;
      uint32_t rtp_ip;
      uint16_t rtp_port;
      char alias[H323_TAM_ALIAS];
      uint64_t begin;   // first packet ts
      uint64_t end;     // last packet ts
      uint32_t npack;
      uint32_t nbytes;
} CallInfoH323;
```

- **h225_ip:** Dirección IP empleada por el protocolo H.225.
- **h225_port:** Puerto empleado por H.225.
- **h245_ip:** Dirección IP empleada por el protocolo H.245.
- **h245_port:** Puerto empleado por H.245.
- **rtp_ip:** Dirección IP empleada por el protocolo RTP.
- **rtp_port:** Puerto empleado por RTP.
- **alias:** Identificador del usuario. Nombre, e-mail, número de teléfono...
- **begin, end:** Timestamp del primer y último paquete recibido.
- **npack:** Número de paquetes recibidos.
- **nbytes:** Número de bytes recibidos.

4.3. Filtrado y detección del tráfico

Para poder capturar las llamadas es necesario filtrar el tráfico perteneciente a los protocolos H.225, H.245 y RTP.

El tráfico H.225 puede ser filtrado por el puerto de destino, ya que el protocolo dispone un puerto específicos, que es el 1720.

Sin embargo el tráfico H.245 no dispone de puertos específicos ya que los puertos empleados son negociados mediante el protocolo H.225 antes de comenzar a enviar paquetes H.245. Debido a esto es necesario pasar todos los paquetes al módulo de

análisis de tráfico, dónde una función primeramente comprueba si pertenece a alguna llamada activa buscando en la tabla hash. Se emplea como función hash para calcular el índice la suma de las direcciones IP. Si pertenece a alguna llamada activa entonces se verifica si es alguno de los mensajes que nos interesan, como veremos más adelante, solamente deberemos detectar si se trata de un mensaje de tipo OpenLogicalChannelAck. Es en este tipo de mensajes donde se envía la información necesaria para establecer el canal de comunicación RTP. Si por el contrario no se encuentra ninguna entrada en la tabla hash o el mensaje no es de tipo OpenLogicalChannelAck, el paquete es descartado.

Una vez tenemos los puertos y direcciones IP empleados para transmitir el flujo RTP podemos pasarle la información mediante una sesión de tipo RTPSession al módulo de captura de tráfico RTP, que se encargará de capturar los flujos de audio, filtrando el tráfico que se debe analizar de una forma similar a como se hace con el correspondiente al protocolo H.245.

4.4. Análisis del tráfico

Una vez delineada la forma en que se detecta el tráfico que se debe analizar, vamos a presentar el mecanismo para analizar el tráfico filtrado en el módulo anterior con la intención de poder extraer la información de cada llamada.

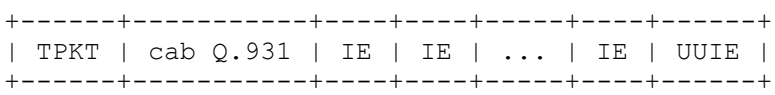
En primer lugar vamos a describir aquellos mensajes involucrados en la llamada H.323 que son empleados para obtener la información relativa a cada llamada. Se realizará una disección de estos mensajes indicando los diferentes campos de información que nos resultan útiles para nuestro propósito y como son decodificados. Posteriormente se pasa a describir el funcionamiento de la máquina de estados utilizada para controlar el estado de las diferentes llamadas activas.

4.4.1. Extracción de datos de la llamada

Para analizar el protocolo H.323, es necesario analizar los mensajes pertenecientes al protocolo H.225 y H.245.

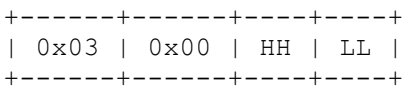
Mensajes H.225

Los mensajes del protocolo H.225 tienen la siguiente estructura:

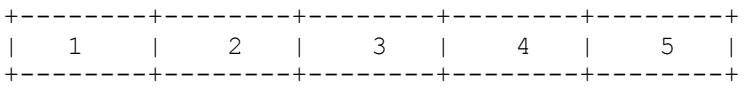


Todos los mensajes comienzan por una cabecera TPKT, a continuación le sigue una cabecera Q.931, seguida de uno o varios IEs (*Information Elements*). Por último un elemento de tipo UUIE (*User-User Infromation Element*).

- Cabecera TPKT: Los primeros 4 bytes pertenecen a la cabecera TPKT, en ella, los dos primeros bytes indican que se trata del protocolo H.225 (código hexadecimal 0x03 y 0x00), los dos siguientes bytes HH y LL representan la longitud del mensaje incluyendo la propia cabecera TPKT en notación *network byte order*.



- Cabecera Q.931: Siguiendo a la cabecera TPKT, los siguientes 5 bytes pertenecen a la cabecera del protocolo Q.931.

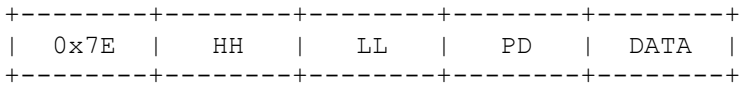


El primer byte se utiliza como discriminador del protocolo, permite conocer que el protocolo que le sigue es el H.225. El siguiente byte indica la longitud del Call Reference Value (valor único de cada llamada, empleado por Q.931) y los siguientes dos bytes son el CRV en *network byte order*. El último byte, es el más interesante, pues muestra de que tipo de mensaje se trata, siendo, los más importantes:

Tipo de mensaje	**Código**
Setup	0x05
Call proceeding	0x02
Alerting	0x01
Connect	0x07
Facility	0x6A
Release complete	0x5A

- IEs: Tras la cabecera Q.931 vienen los diferentes IEs, pueden ser uno o varios, varían en función del tipo de mensaje (setup, alerting...), conteniendo cada uno diferentes elementos de información ("Calling Party Number", "Called Party Number", "Display"...).

- UUIE: Por último está el UUIE (User-User Information Element) que tiene la siguiente estructura:



El primer byte es un discriminador, sirve para saber cuándo comienza el objeto UUIE. HH y LL es la longitud de DATA en notación little-endian y PD es un

discriminador de protocolo para ASN.1 (0x05).

Data contiene el objeto H323-UserInformation, codificado mediante las PER en ASN.1. Este objeto contiene los identificadores empleados por los participantes de la llamada, así como las direcciones IP y los puertos empleados. De aquí puede también extraerse, aunque no es una información obligatoria, si el canal H.245 irá tunelizado o en un canal aparte.

En la siguiente imagen se puede ver en una captura de Wireshark un ejemplo de mensaje CONNECT.

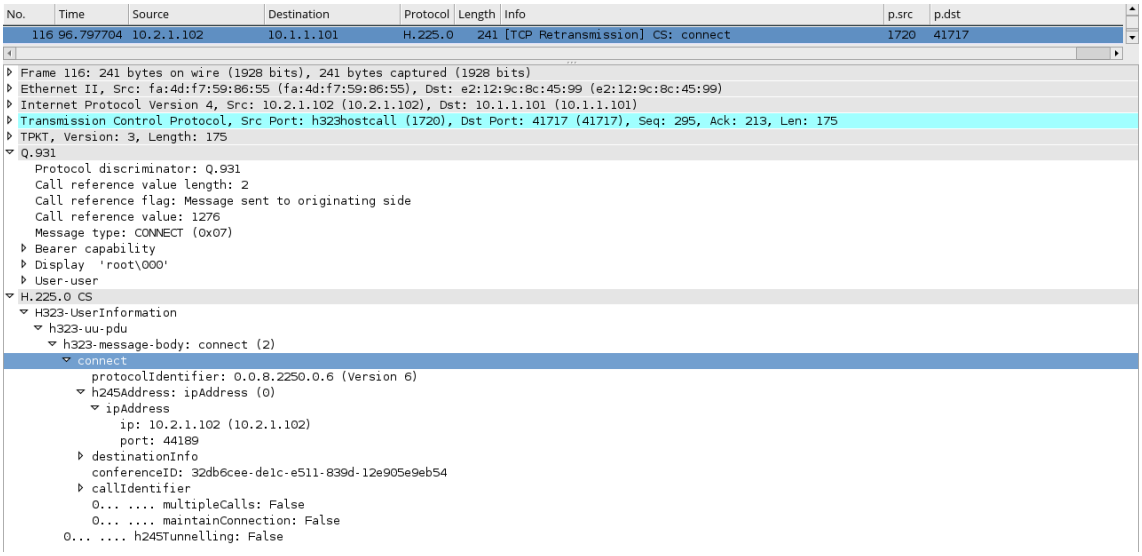


Ilustración 10

Ejemplo de mensaje Connect

Mensajes H.245

Los mensajes H.245, tienen la siguiente forma:

```
+------+-----------+-----------+---+-----------+-----------+
| TPKT | H.245 PDU | H.245 PDU |...| H.245 PDU | H.245 PDU |
+------+-----------+-----------+---+-----------+-----------+
```

- Cabecera TPKT: Los primeros 4 bytes pertenecen a la cabecera TPKT, exactamente igual que en los mensajes H.225.

- Elementos H.245PDU: Siguen a la cabecera TPKT. Normalmente se envía un único elemento H.24PDU por mensaje, no obstante, puede haber varios en un mismo mensaje. Estos elementos están codificados en ASN.1, por lo que antes de poder interpretarlos es necesario que se decodifiquen.

Nosotros solo estamos interesados en los mensajes de tipo openLogicalChannelAck, es decir, aquellos que se emplean como respuesta a una petición de apertura de un canal lógico. Es en este tipo de mensajes donde se envía la información necesaria para establecer el flujo de audio entre los dos terminales de la comunicación. Este mensaje también suele incluir los parámetros necesarios para abrir un canal de control para RTP (RTCP). Cada canal lógico se identifica con un SessionID único.

Los canales abiertos son unidireccionales, por lo que hay que tener en cuenta que es necesario detectar dos mensajes de este tipo para capturar la conversación completa, es decir en ambas direcciones. También recordar que al ser canales diferentes no tienen por qué compartir códecs, por lo que hay que capturar también las capacidades negociadas por los terminales para poder determinar el códec utilizado.

La imagen muestra una captura de Wireshark en la que se puede ver un mensaje de tipo OpenLogicalChannelAck.

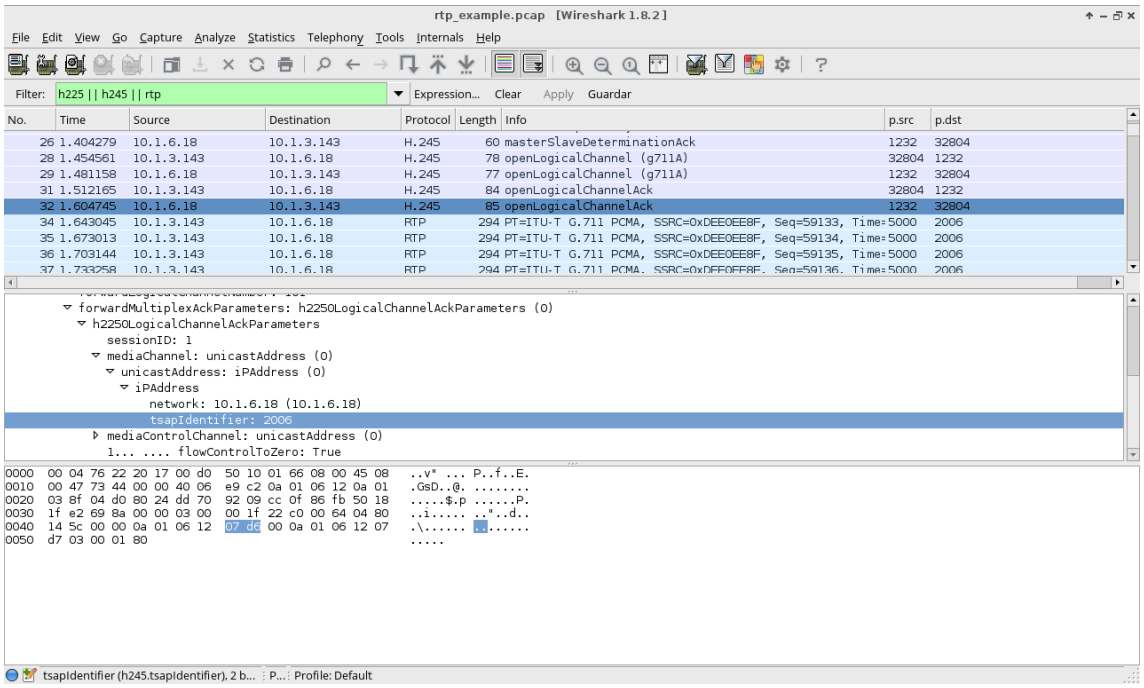


Ilustración 11

Ejemplo de mensaje OpenLogicalChannelAck

4.4.2. Máquina de estados y control de estado de la conexión

Para el diseño de la máquina de estados nos basamos en los diferentes estados en los que puede estar una llamada H.323.

Tenemos los siguientes estados:

```
#define READY 0
#define SETUP_RECEIVED 1
#define CALL_PROCEDING_RECEIVED 2
#define ALERTING_RECEIVED 3
#define CONNECT_RECEIVED 4
#define CONNECTED 5
```

Las transiciones entre los distintos estados se establecen según el mensaje recibido. Para ello definimos los mensajes "interesantes" que podemos recibir:

```
/* Tipos de mensaje */
#define SETUP 0x05
#define CALLPROCEEDING 0x02
#define ALERTING 0x01
#define CONNECT 0x07
#define RELEASE 0x4D
#define RELEASECOMPLETE 0x5A
#define FACILITY 0x6A
```

A continuación, en el siguiente diagrama se pude ver de una forma gráfica la definición de la máquina de estados:

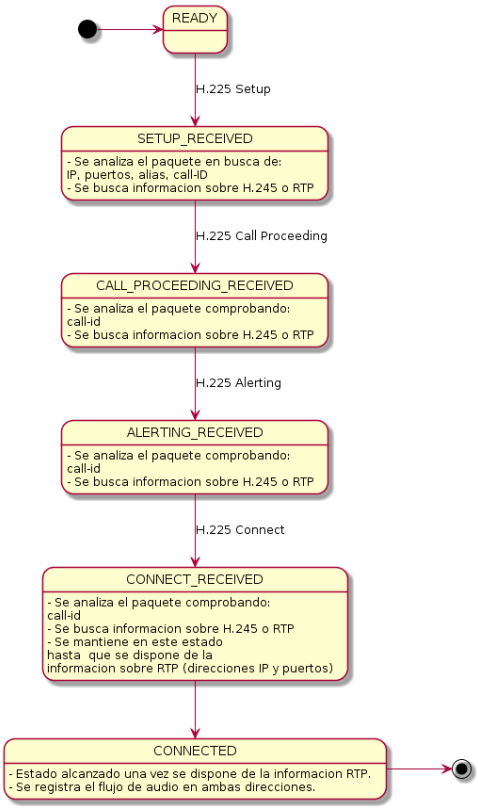


Ilustración 12

Máquina de estados

4.5. Particularidades de la implementación

Una de las partes más complejas de la implementación ha sido encontrar un decodificador válido para ASN.1. Se probaron varios compiladores sin resultados satisfactorios pero finalmente logramos que la empresa OSS Nokalva nos prestase una licencia para poder ejecutar una versión de prueba durante el desarrollo del TFG. Gracias a ello pudimos decodificar los mensajes para luego poder interpretarlos.

Otra particularidad destacada fue que necesitábamos poder indexar las llamadas de una forma que pudiesen ser localizadas en la tabla hash al encontrar un paquete H.225 o H.245. La única información que comparten estos dos protocolos son las direcciones IP de origen y destino, por lo que fue necesario definir una función hash que tuviera en cuenta únicamente estos datos como entrada.

Como esta función tiene un espacio de llegada reducido frente al total de posibles combinaciones de la suma de dos valores enteros no signados representados con 32 bits, se evaluó el impacto de las colisiones, para evitar degradaciones del rendimiento del módulo. Para ello, realizamos una evaluación tanto analítica como empírica de las propiedades de la función hash utilizada, que viene dada por

$$f(d_1, d_2) = mod(d_1 + d_2, size_{table})$$

Donde d_1 y d_2 son las direcciones IP (como se ha comentado anteriormente, enteros no signados de 32 bits) y $size_{table}$ es el número de entradas de la tabla hash. En este contexto, comprobamos tanto analíticamente como empíricamente que si $size_{table}$ es una potencia de 2, el hash resulta ser uniformemente distribuido (al considerar todas las posibles combinaciones de d_1 y d_2).

4.6. Conclusiones

Para la implementación del módulo se ha intentado seguir un diseño simple, que permita manejar de forma eficiente la información relativa a cada llamada. Además se ha utilizado de un diseño modular que mantiene la independencia de los diferentes módulos funcionales separando la parte de captura y filtrado del tráfico, de la decodificación y de su posterior análisis.

El filtrado del tráfico H.225 se realiza discriminando los paquetes por el número de puerto que establece el protocolo, el 1720. Para H.245 es necesario pasar los paquetes al módulo de análisis y detectar si se trata de un mensaje H.245 mediante otros mecanismos más complejos.

En el módulo de análisis se controla el estado de cada llamada mediante una máquina de estados que transita en función del último mensaje recibido.

Hay que destacar los numerosos problemas con la decodificación de ASN.1, que se han resuelto proporcionando una estructura adecuada para que en un futuro la implementación pueda modificarse fácilmente, cumpliendo el requisito que se estableció para minimizar riesgos derivados de este tipo de situaciones.

5. Validación

5.1. Introducción

En este capítulo se presentan las pruebas que se ha efectuado sobre el software implementado para este TFG. A lo largo del desarrollo del proyecto se han realizado diversas pruebas unitarias para comprobar que los diferentes procedimientos y módulos implementados trabajaban de la forma adecuada. Sin embargo, las pruebas realmente relevantes son las pruebas de validación, que son en base a las cuales podemos decidir si el proyecto se ajusta a los requerimientos iniciales descritos en el Capítulo 3 y, por lo tanto, podemos considerarlo como un éxito; o por el contrario, no cumple estos requisitos, en este caso será necesario analizar qué ha fallado y tomar medidas en consecuencia con la finalidad de subsanar el fallo o si no es posible, minimizar sus consecuencias.

En primer lugar, y con el fin de garantizar que las pruebas sean repetibles y muestren el buen funcionamiento del módulo desarrollado, se describen los distintos escenarios en los que ha sido validado. Posteriormente, a partir del análisis de los resultados obtenidos se exponen las conclusiones extraídas tras la realización de las pruebas de validación.

5.2. Montaje y configuración del entorno de pruebas

En este apartado se explica cómo están dispuestos los diferentes escenarios bajo los que se realizan las pruebas de validación. En primer lugar, se describen los elementos y sistemas empleados, y posteriormente se describen los escenarios indicando cómo se distribuyen dichos elementos.

El sistema VoIPCallMon se puede ejecutar sobre cualquier máquina con sistema operativo basado en Linux (preferiblemente de la familia Debian). Para la evaluación, se ha utilizado una máquina con las especificaciones técnicas de estos equipos que se detallan a continuación:

Procesador	Intel Pentium D, CPU 2x 2.80GHz
Memoria	3 GBytes
Disco duro	120 GBytes
Tarjeta de red	Puerto Ethernet compatible con RJ45 10/100/1000 Mbits/s
Sistema operativo	Ubuntu 12.04

Para simular máquinas virtuales se utiliza el software de virtualización que provee VMware [23], VMware Player. El sistema operativo emulado en las máquinas virtuales es una distribución de derivada de Ubuntu, Lubuntu.

A continuación se describen los elementos utilizados para transformar la red en un sistema VoIP.

Gatekeeper

En cuanto al software empleado para transformar los equipos en un sistema de VoIP H.323, se ha optado por escoger alternativas con licencia de código libre. En el caso de no existir esta alternativa, se ha tratado de utilizar herramientas libres de coste.

Tras experimentar con varias opciones, finalmente decidimos emplear como software para simular la centralita contra la que se registran los diferentes terminales integrantes de la red H.323, GNU Gatekeeper (GnuGk). GnuGk es un programa con licencia GPL, multiplataforma, que permite controlar la red H.323 y provee una serie de funciones auxiliares, como son: la de añadir seguridad y autenticación para los diferentes participantes, autorización de llamadas, desvío y contestador automático de llamadas... La versión empleada ha sido la 3.4.0.

Para su configuración empleamos el fichero "gatekeeper.ini" que se muestra a continuación:

```
[Gatekeeper::Main]
Fourtytwo=42
name=gnugk
[GkStatus::Auth]
rule=allow
```

GnuGk tiene dos modos principales de funcionamiento: modo directo y modo enrutado. En el modo directo, los mensajes de señalización de la llamada (protocolo H.225) se intercambian directamente entre ambos participantes, por lo que el gatekeeper sólo actúa para registrar a los participantes y como localizador de los diferentes puntos terminales. En el modo enrutado, estos mensajes, son enviados a través del gatekeeper. En este modo es posible configurar el gatekeeper para que los mensajes del protocolo H.245 también sean enviados a través del gatekeeper.

Para lanzar el programa basta con ejecutar en la terminal:

- Modo directo: `~$ gnugk -d`
- Modo enrutado: `~$ gnugk -r`

  `~$ gnugk -rr` Si se desea enrutar también el protocolo H.245

Al comando anterior se le puede añadir la opción -t para mostrar información, con distintos niveles de verbosidad configurables. Esta opción resulta muy útil para depurar los posibles errores derivados de una mala configuración.

Terminales

Como terminales se han utilizado dos softphones: ekiga y simpleopal. Ekiga es un softphone multiplataforma utilizado en entornos reales, con licencia de código abierto, y soporte para varios protocolos (entre ellos H.323).

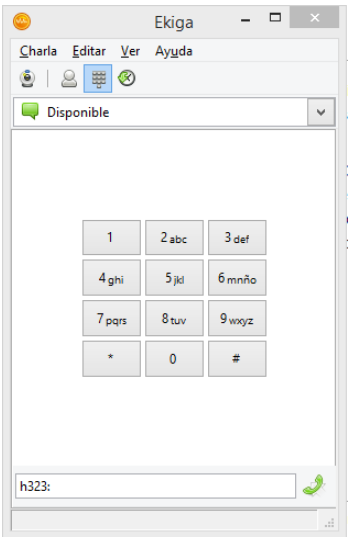


Ilustración 13

Captura de pantalla del programa Ekiga

Simpleopal sin embargo, es más adecuado para realizar pruebas de conectividad, resultando particularmente útil en las primeras pruebas de funcionamiento. También se distribuye como software de código abierto. La principal ventaja que presenta frente a Ekiga es que permite configurar muchos más parámetros de la comunicación H.323: algunos ejemplos son el empleo del método fastconnect para establecer la llamada, o el tunelizado de H.245 bajo H.225. Estas opciones son muy útiles para probar el programa bajo diversas situaciones y comprobar que el disector es capaz de analizar los diferentes tipos de llamada H.323.

Algunas opciones adicionales de simpleopal son:

- -f, desactiva faststart
- -T desactiva la tunelización de H.245, es decir hace que H.245 se envíe con puertos diferentes a H.225.
- -e se utiliza para evitar la supresión de silencios. Esta opción es necesaria para nuestras pruebas, ya que al estar trabajando en entornos virtuales sin entradas de audio, sin ella no se enviaría tráfico RTP.

Ejemplo de ejecución de simpleopal:

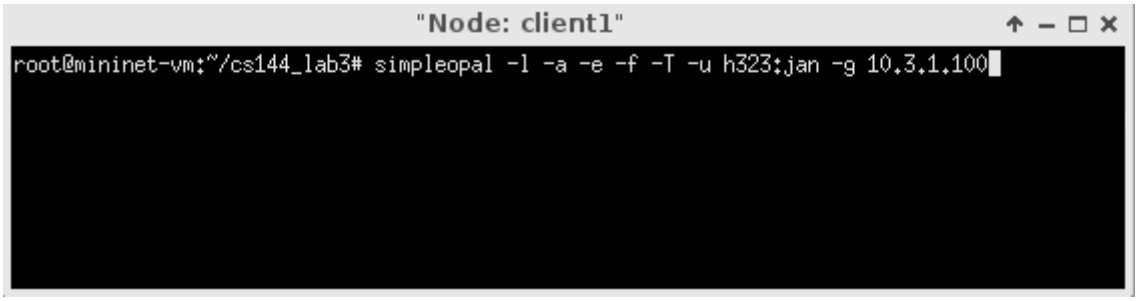


Ilustración 14

Ejemplo de ejecución de Simpleopal en modo escucha

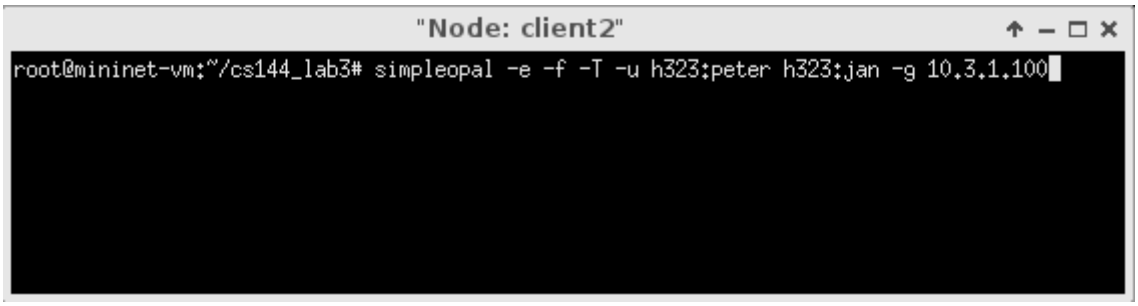


Ilustración 15

Ejemplo de ejecución de Simpleopal en modo llamada

Tanto Ekiga, como Simpleopal, como Gnugk pueden ser obtenidos de los repositorios oficiales de Ubuntu.

Descripción de escenarios

A continuación describimos los diferentes escenarios de evaluación del software desarrollado a los largo de este TFG.

Escenario 1: prueba en el entorno de desarrollo

Esta prueba tiene el objetivo de comprobar que el módulo implementado es capaz de detectar una llamada H.323 y obtener los flujos de audio RTP de la conversación.

Se ejecuta en el entorno de desarrollo junto con una máquina virtual. En la máquina virtual se ejecuta el gatekeeper junto con uno de los softphones. En el PC host se ejecuta el otro softphone.

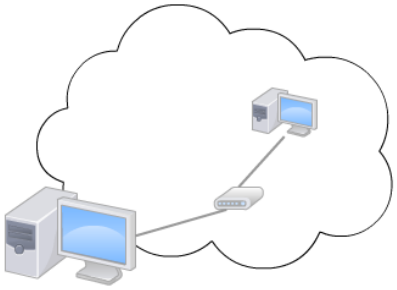


Ilustración 16

Escenario 1

Escenario 2: prueba en entorno virtualizado

Para probar el software en un entorno más realista se usó un entorno de virtualización de redes. Este tipo de entornos permiten virtualizar de una forma bastante ligera una red con diferentes host conectados. La mayoría de entornos permiten configurar la red de acuerdo a las necesidades de cada usuario, especificando los parámetros de la red en un fichero de configuración.

Se optó por escoger mininet como entorno de virtualización debido a que reúne todas las características necesarias para llevar a cabo el tipo de pruebas que permiten validar nuestra herramienta.

Mininet es un simulador de red que permite virtualizar de manera muy ligera una serie de hosts, switches, enlaces y routers en una sola máquina. Mediante ficheros de configuración escritos en Python es posible definir la topología y los diferentes elementos de la red emulada.

La red desplegada tiene la siguiente configuración:

- 7 hosts terminales.
- 1 PBX, es otro host más que ejecuta GnuGk.
- 2 switches, interconectando los hosts.
- 1 router que divide la red en 3 subredes.

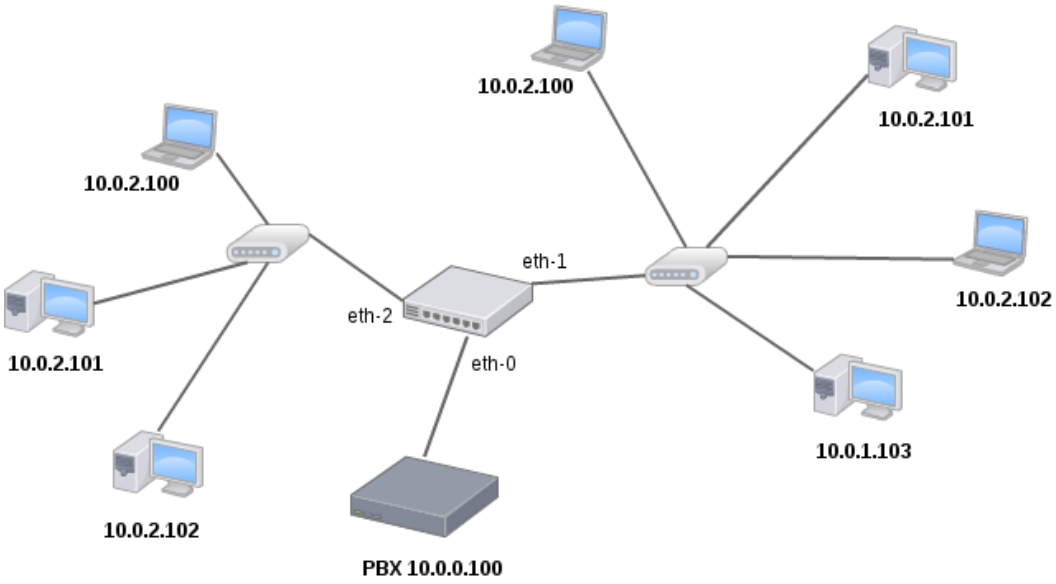


Ilustración 17

Escenario 2

5.3. Pruebas realizadas

Tras describir los escenarios en los que se validará el sistema, pasamos a describir el transcurso de las pruebas y los resultados obtenidos tras su realización.

5.3.1. Pruebas de validación

Para realizar la validación, previamente ha sido necesario realizar capturas de tráfico mediante la herramienta Wireshark. En este sentido, hemos planteado las pruebas del módulo en la forma de análisis forense de tráfico de red, ya que sólo queremos comprobar el funcionamiento y la correcta extracción de los datos de las llamadas cursadas. Así, posteriormente se pasa esa captura de tráfico como parámetro de entrada a VoIPCallMon, lo que permite trabajar y repetir las pruebas en el caso de detectar algún error en el funcionamiento del programa.

```
#!/bin/bash

MODE=1
INPUT_FILE=xxxxxxx.pcap
SIGNALING_EXP=300
RTP_EXP=15
OUTPUT_PATH=../capturas_paquetes/resultados
COLEC_RAW=0  #false
INPUT_PATH=../capturas_paquetes
INPUT_FORMAT=pcap

#Para ejecutar RTPTracker:
./RTPTracker $MODE $INPUT_FILE $SIGNALING_EXP $RTP_EXP $OUTPUT_PATH $COLEC_RAW $INPUT_PATH $INPUT_FORMAT
```

Ilustración 18

Script utilizado para ejecutar VoIPCAllMon

En la siguiente captura se muestra un ejemplo de ejecución del programa VoIPCallMon, con la salida y las trazas que proporciona esta herramienta de análisis:

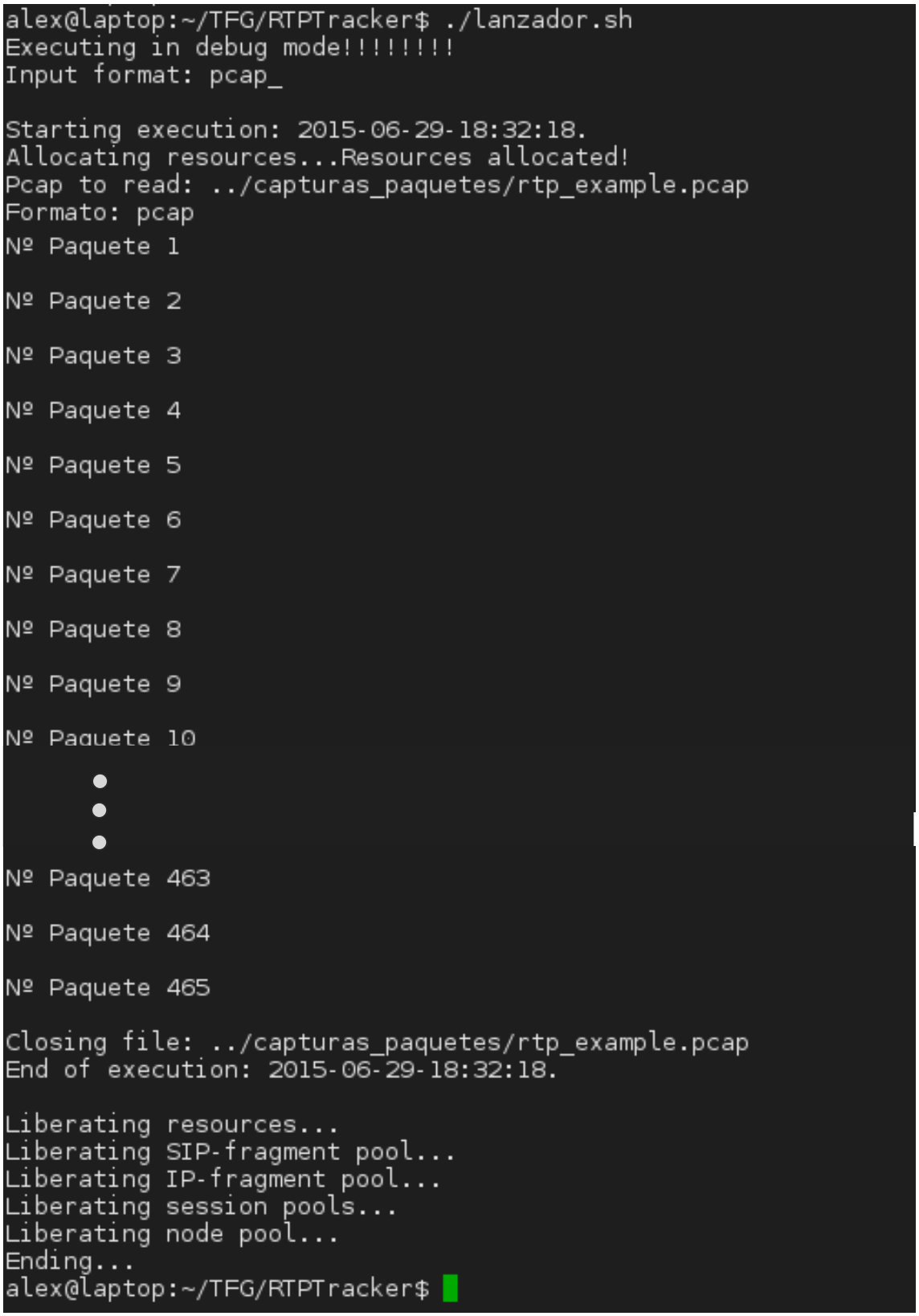


Ilustración 19

Ejecución de VoIPCallMon

La salida del programa genera los directorios RAW y debug, que contienen los ficheros en los que se persisten los datos de las llamadas:

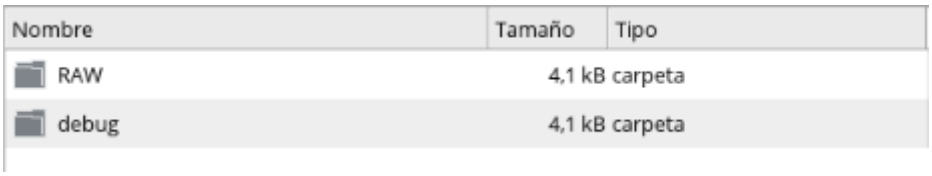


Ilustración 20

Directorios donde se almacena la salida de VoIPCallMon

En particular, en el directorio debug se almacenan los ficheros con los registros generados para las llamadas cursadas con cada uno de los protocolos soportados.

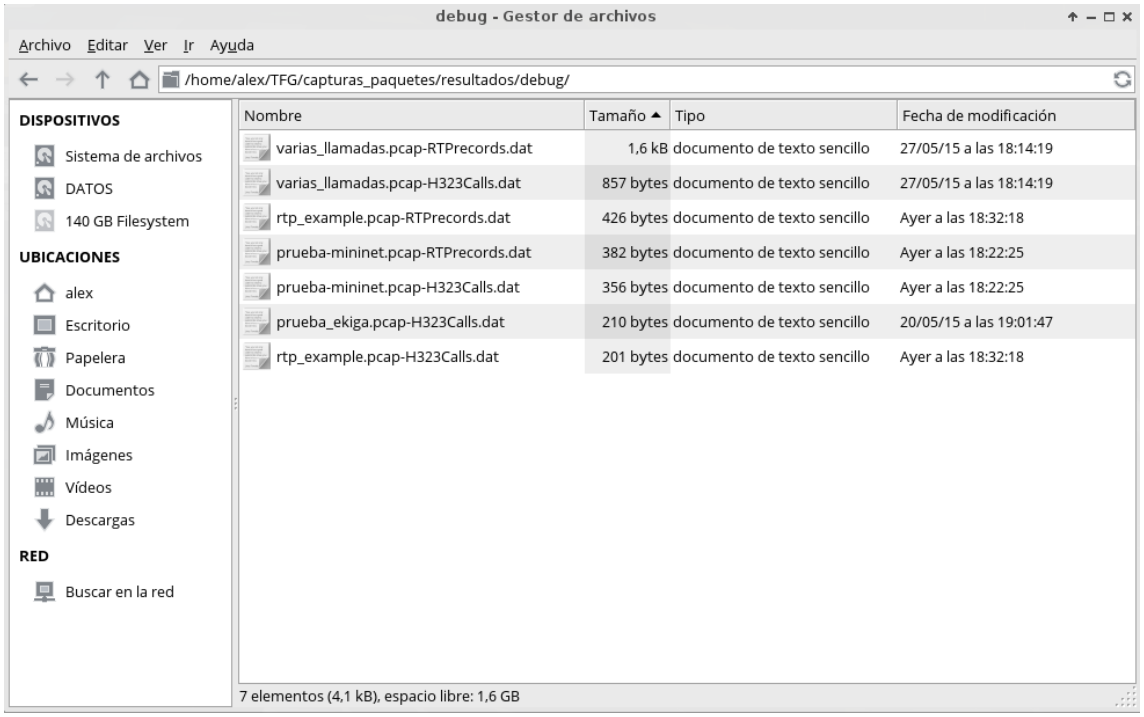


Ilustración 21

Registros de llamadas generados

En el directorio RAW se almacenan las capturas de tráfico RTP en formato pcap. Estas se organizan en subcarpetas nombradas según el timestamp del primer paquete RTP recibido. Cada fichero .pcap contiene el tráfico que se corresponde con el audio transmitido en una única dirección, por lo que para cada llamada se generan ficheros que deben ser mezclados si se desean observar los flujos de datos en ambos sentidos.

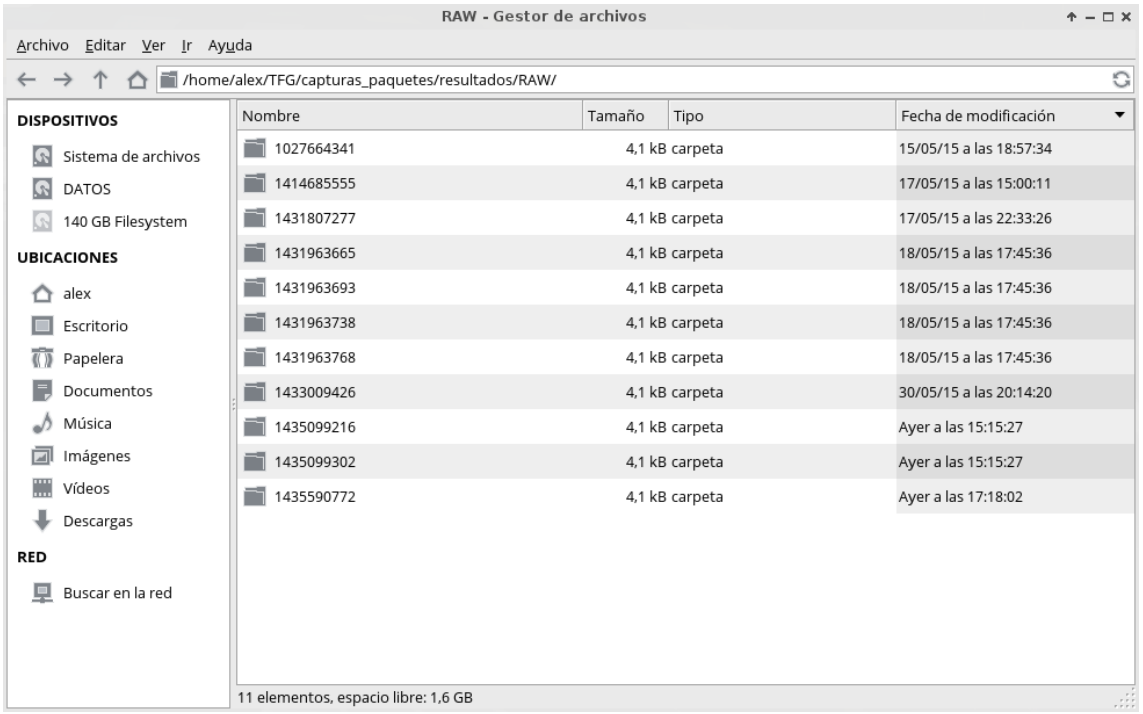


Ilustración 22

Directorios donde se almacenan las capturas de audio RTP

Escenario 1: prueba en el entorno de desarrollo

En primer lugar se va a realizar una llamada H.323 convencional entre dos participantes; En segundo lugar, se realiza una llamada que termina con un código de error, y por último una llamada con el protocolo H.245 tunelizado en H.225.

En la siguiente captura se puede ver los ficheros generados como salida del programa.

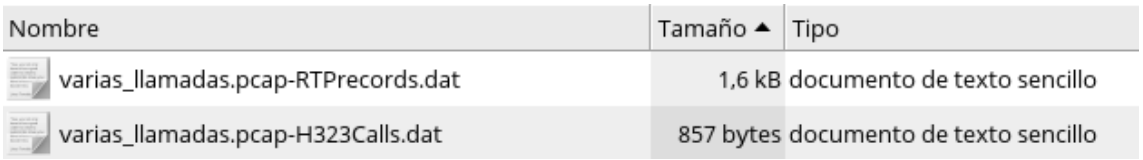


Ilustración 23

Ficheros generados como salida de VoIPCallMon

En el fichero de registros H.323 se persisten los datos de las llamadas detectadas que se han realizado junto con algunos parámetros de la comunicación:

- Timestamp del primer y último paquete recibido.
- Call-ID de la llamda.
- Direcciones IP y puertos usados para el tráfico de señalización de llamada, H.225.
- Tiempo en segundos transcurrido de conversación.
- Código de finalización de la llamada.
- Códecs empleados para el audio.
- El último campo indica si el protocolo H.245 va tunelizado junto con H.225 o en un canal aparte.

Se muestra el fichero dividido en dos para poder facilitar su lectura. El fichero completo corresponde a concatenar la segunda parte a la primera.

1431963665440813	1431963674346911	2680c6f6e1fbe411970c0016ea53e766	45165	1720
1431963693958935	1431963705085325	f622c507e2fbe41187000016ea53e766	37719	1720
1431963738659307	1431963751414343	8ad86a22e2fbe41190920016ea53e766	44414	1720

Ilustración 24

Parte izquierda del fichero de resultados

192.168.1.128	192.168.1.130	peter	jan	8.906098	8.903970	200	200	g711	g711	FALSE
192.168.1.128	192.168.1.130	jacobo	jan	11.125794	11.126202	-1	-1	g711	g711	FALSE
192.168.1.128	192.168.1.130	jan julian		12.755036	12.752843	200	200	g711	g711	TRUE

Ilustración 25

Parte derecha del fichero de resultados

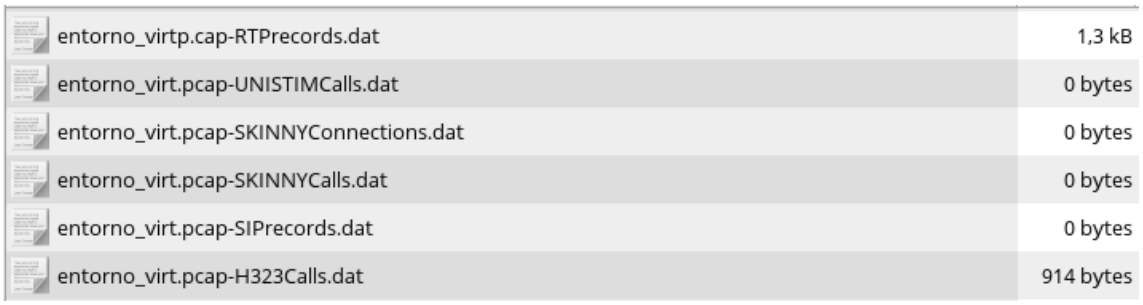
El fichero de registros RTP incluye la información referente a la transmisión del flujo de audio. Estos datos son:

- Timestamp del primer y último mensaje RTP recibido.
- Direcciones IP y puertos usados por ambos participantes para RTP.
- El número de paquetes transmitidos en una y otra dirección.
- El número de bytes transmitidos en una y otra dirección.
- El ratio de paquetes perdidos en una y otra dirección.
- El tamaño medio de cada paquete, en bytes.
- La duración de la llamada.
- Los ficheros en los que se almacenan los paquetes de audio capturados.

Se observa que los resultados obtenidos son los esperados, en base a lo esperado por la definición de la prueba.

Escenario 2: prueba en entorno virtualizado

En este entorno, vamos a probar a realizar varias llamadas entre los diferentes integrantes de la red. A continuación se muestran los resultados obtenidos.



entorno_virtp.cap-RTPrecords.dat	1,3 kB
entorno_virt.pcap-UNISTIMCalls.dat	0 bytes
entorno_virt.pcap-SKINNYConnections.dat	0 bytes
entorno_virt.pcap-SKINNYCalls.dat	0 bytes
entorno_virt.pcap-SIPrecords.dat	0 bytes
entorno_virt.pcap-H323Calls.dat	914 bytes

Ilustración 26

Ficheros generados como salida de VoIPCallMon

Se han realizado 6 llamadas diferentes. A continuación, en las siguientes imágenes se pueden observar los resultados. Como en el apartado anterior, el fichero se muestra partido en 2 para facilitar su lectura.

```
1435698974359182    1435698997327094    34f1cce7da1de51185ca0016ea53e766    34265   1720
1435698826358430    1435698840697105    32a1958fda1de5119dc30016ea53e766    41334   1720
1435698760335254    1435698773164917    18c13968da1de5119f800016ea53e766    34984   1720
1435698882855144    1435698908521908    066342b1da1de51187570016ea53e766    49852   1720
1435699054530879    1435699054648914    a65d9617db1de51181920016ea53e766    45613   1720
1435699108511037    1435699127461191    e238c337db1de51183140016ea53e766    34359   1720
```

Ilustración 27

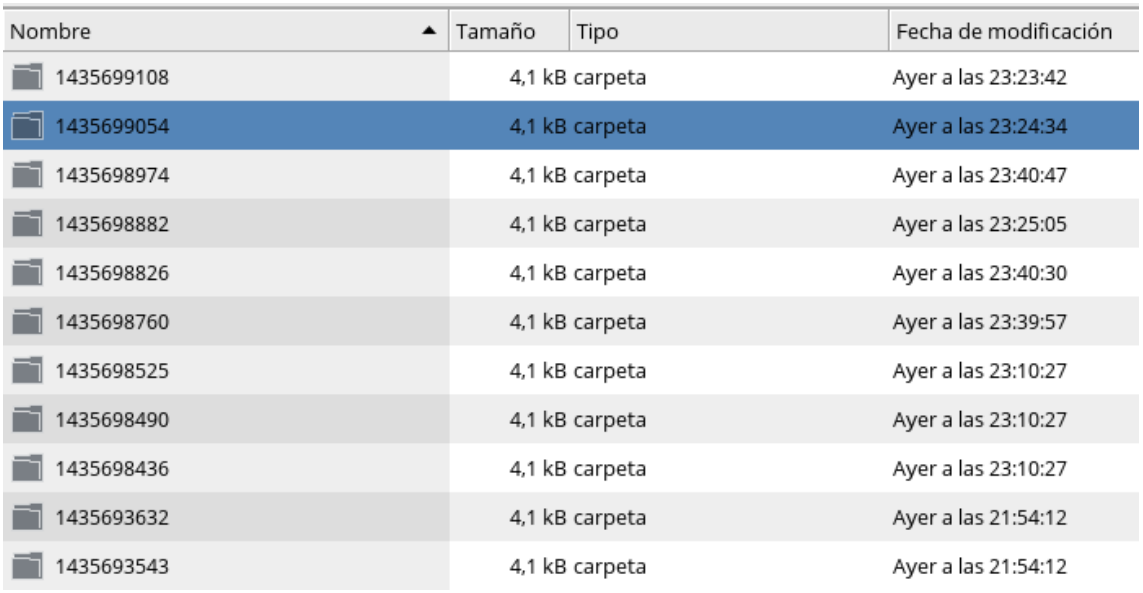
Parte izquierda del fichero de resultados

```
10.1.2.101  10.1.1.103  malcom  jan 22.967886   22.967884   200 200 g711    g711    FALSE
10.1.2.100  10.1.1.102  alice   jan 14.337626   14.338632   -1  -1  g711    g711    FALSE
10.1.1.101  10.1.2.103  peter   jan 12.829620   12.829565   200 200 g711    g711    TRUE
10.1.2.100  10.1.1.102  bob     jan 25.666744   25.666726   200 200 g711    g711    FALSE
10.1.2.101  10.1.1.103  bender  jan 0.118035    0.109091    -1  -1  g711    g711    TRUE
10.1.2.101  10.1.2.102  mary    jan 18.950146   18.950128   200 200 g711    g711    TRUE
```

Ilustración 28

Parte derecha del fichero de resultados

De acuerdo a las condiciones de prueba definidas, estos resultados permiten concluir que todas las llamadas han sido detectadas correctamente.



Nombre	Tamaño	Tipo	Fecha de modificación
1435699108	4,1 kB	carpeta	Ayer a las 23:23:42
1435699054	4,1 kB	carpeta	Ayer a las 23:24:34
1435698974	4,1 kB	carpeta	Ayer a las 23:40:47
1435698882	4,1 kB	carpeta	Ayer a las 23:25:05
1435698826	4,1 kB	carpeta	Ayer a las 23:40:30
1435698760	4,1 kB	carpeta	Ayer a las 23:39:57
1435698525	4,1 kB	carpeta	Ayer a las 23:10:27
1435698490	4,1 kB	carpeta	Ayer a las 23:10:27
1435698436	4,1 kB	carpeta	Ayer a las 23:10:27
1435693632	4,1 kB	carpeta	Ayer a las 21:54:12
1435693543	4,1 kB	carpeta	Ayer a las 21:54:12

Ilustración 29

Directorios donde se almacenan las capturas de audio RTP

5.4. Conclusiones

Las pruebas efectuadas nos han servido para comprobar que el software desarrollado cumple con los requisitos funcionales que habíamos descrito en el Capítulo 3.

Esta evaluación se ha realizado bajo diferentes escenarios controlados: un primer escenario muy simple en el entorno de desarrollo y posteriormente en un entorno de red virtualizada, basado en mininet. Los entornos virtualizados de red abren la puerta a la realización de este tipo de pruebas, con bajo coste y alta flexibilidad a la hora de definir escenarios. Este experimento ha permitido comprobar la adecuación de mininet para plantear entornos de pruebas para herramientas como la que hemos desarrollado, lo que puede resultar de interés tanto para profesionales como para investigadores del área de redes. Desde un punto de vista más pedagógico, la inclusión de este elemento en el TFG ha permitido que se realice además una pequeña aproximación a este tipo de herramientas, cada día más extendidas con la expansión de las Redes Definidas por Software (SDN por sus siglas en inglés).

Finalmente, estas pruebas han permitido detectar algunas limitaciones de la librería empleada para la decodificación de los elementos enviados utilizando ASN.1. Se ha observado que existen algunos problemas con la decodificación de elementos incluidos en los mensajes faststart de transmitidos por ekiga, pero la sustitución de esta librería está incluida como trabajo futuro.

6. Conclusiones

6.1. Resumen

Este trabajo muestra una visión completa del proceso de desarrollo de un módulo para monitorización y análisis de tráfico de despliegues de VoIP basados en el protocolo H.323.

Para empezar, hemos proporcionado una visión general de todos los elementos involucrados en las instalaciones de VoIP. Se han descrito sus elementos más importantes, así como, los protocolos de señalización de llamada más utilizados y algunas cuestiones relativas a la gestión de estos servicios. Como se ha comentado a lo largo del trabajo, las redes IP tienen una serie de peculiaridades que hacen indispensable una monitorización y gestión de la red para poder lograr unos parámetros óptimos de calidad en la transmisión audio en tiempo real. En este sentido, hemos listado algunos parámetros que se emplean para medir la calidad en este tipo de redes, ya que es un aspecto crítico de este tipo de servicios y una de las motivaciones del sistema VoIPCallMon en el que se integra el software desarrollado.

Centrándonos en el desarrollo de un elemento funcional para este sistema, hemos explicado con mayor detalle el protocolo de señalización de llamadas H.323 con la finalidad de introducir al lector en el tipo de tráfico en el cual se va a centrar el software desarrollado durante este TFG. Además, las características de este protocolo y la estructura de sus mensajes son las que, en cierta medida, han impuesto ciertas restricciones al diseño de la solución propuesta.

En ese sentido, se ha proporcionado una descripción completa de las distintas fases de desarrollo del módulo de análisis de H.323. Se han definido una serie de requisitos completos que han permitido evaluar el buen funcionamiento del módulo desarrollado. Resumiendo, las principales características de este son:

- Detección de llamadas bajo el protocolo de señalización H.323, independientemente de si el protocolo de control H.245 va tunelizado en el protocolo H.225 o en un canal separado.

- Aunque la detección de llamadas faststart no es posible, sí que se ha implementado la lógica necesaria para el control de este tipo de llamadas, por lo que si se resuelve el problema de la decodificación de los elementos faststart, estas llamadas serían soportadas.

- Recolección de información acerca de las llamadas detectadas, entre esta información está: las direcciones IP y puertos utilizados por los participantes de la conversación para el tráfico de señalización y para RTP, parámetros relevantes de la conexión de señalización y otros parámetros utilizados para evaluar la calidad de la conexión RTP, como la duración de la llamada, número de paquetes transmitidos…

6.2. Principales aportaciones

En primer lugar, este TFG ha permitido ampliar la funcionalidad del sistema VoIPCallMon, incluyendo el soporte para llamadas cursadas con H.323. Este protocolo, ya antiguo pero aún muy popular en despliegues operativos de VoIP, reviste una complejidad adicional por la forma en que los campos de datos son codificados en los mensajes. Este módulo permitirá, por tanto, extender el rango de aplicación de la herramienta con fines bien comerciales bien relacionados con la investigación en el área de gestión de red.

No obstante, hay una serie de aportaciones adicionales de este TFG, quizás menos tangibles pero no por ello menos importantes. En primer lugar, se ha realizado una evaluación de las herramientas para decodificación de ASN.1. Este trabajo de evaluación ha mostrado la necesidad de generar una nueva librería que pueda ser utilizada en entornos de gestión con requisitos de alta tasa, y que proporcione soporte y funcionalidad completa.

Por otro lado, se han considerado diversas herramientas para la generación de tráfico de VoIP con H.323 como protocolo de señalización. Esta tarea no es del todo sencilla, puesto que no resultó fácil encontrar la configuración más adecuada para conseguir entornos operativos realistas. En muchos casos, estos problemas han estado asociados a la falta de información, ya que la documentación existente en muchos casos no se correspondía con las versiones disponibles del software. Además, la gran variedad de forks de los distintos proyectos (muchos de ellos descontinuados o en situación de semi-abandono) han añadido complejidad a estas tareas. Creemos que esta revisión puede ser útil para otros profesionales que requieran un entorno de pruebas para el desarrollo de nuevas herramientas en esta área.

Finalmente, hemos mostrado cómo es posible definir entornos controlados de prueba utilizando entornos de redes virtualizadas (en nuestro caso, mininet). El trabajo proporciona, por tanto, un ejemplo de uso de estas tecnologías que están adquiriendo relevancia en el mundo de las telecomunicaciones. Con este ejemplo, hemos ilustrado como este tipo de entornos pueden ayudar durante las fases de evaluación de software de monitorización y análisis de red.

6.3. Trabajo futuro

Este proyecto se enmarca dentro de un campo amplio y con cambios cada vez más rápidos, por lo que es posible definir varias líneas de trabajo futuro:

- **Implementación de una librería de decodificación de ASN.1.** La realización de una librería propia de decodificación de mensajes que utilizan ASN.1 y PER es una labor de gran complejidad que excede los límites de este TFG. No obstante, puede resultar interesante, dados los problemas encontrados con las distintas alternativas evaluadas y las limitaciones de licencia de la finalmente incluida.

- **Añadir soporte para la detección de multiconferencias y transmisión en multicast.** Esta funcionalidad no se ha implementado, debido a que no es propia de H.323 sino que se describe en un documento suplementario que añade funcionalidades adicionales al protocolo. Sería necesario evaluar con mayor profundidad este tipo de modos de transmisión, para poder asegurar la correcta gestión de estas situaciones.

- **Realización de pruebas de rendimiento.** Debido a la complejidad asociada a la realización de pruebas de rendimiento completas de este tipo de sistemas, no ha sido posible evaluar cuáles son los límites de rendimiento del módulo desarrollado. Encontrar este límite puede ayudar a mejorar posibles limitaciones en la implementación actual.

7. Referencias

[1] D. W. a. S. Wheeler, «ALT-C 2006: The next generation,» de *ALT-C 2006*, Heriot-Watt University, Edinburgh, Scotland, UK, September 2006.

[2] Y. Y. I. N. H. J. Yongguo Zhao, «IP Telephony - New Horizon for Telemedicine and e-Health,» *Journal of Medical Systems,* vol. 26, 2002.

[3] S. &. P. F. N. Karapantazis, «VoIP: A comprehensive survey on a promising technology,» *Computer Networks,* 2009.

[4] Comisión Nacional de los Mercados y la Competencia, «Infrome Anual 2014,» 2014.

[5] International Telecommunication Union, «TMN management functions,» 02/2000.

[6] J. L. García-Dorado, P. M. Santiago del Río, R. Javier, M. David, V. Moreno, J. Lopez de Vergara y J. Aracil, «Low-cost and high-performance: VoIP monitoring and full data retention at multi-Gb/s rates using commodity hardware,» *INTERNATIONAL JOURNAL OF NETWORK MANAGEMENT,* 2014.

[7] I. A. Tool, *III ASN.1 Tools,* http://iiiasn1.sourceforge.net/.

[8] L. W. <vlm@lionet.info>, *ASN.1 Compiler,* https://lionet.info/asn1c/compiler.html.

[9] OSS Nokalva, «ASN.1 Tools for C,» [En línea]. Available: http://www.oss.com/.

[10] A. R. G. &. B. J. Orebaugh, «Wireshark & Ethereal network protocol analyzer toolkit.,» Syngress, 2006.

[11] U. &. W. E. Lamping, «Wireshark User's Guide,» 2004.

[12] H. S. a. J. Rosenberg, «A Comparison of SIP and H.323 for Internet,» International Workshop on Network and Operating System Support for Digital Audio and Video, Cambridge, England, 1998.

[13] IETF, «RTP: A Transport Protocol for Real-Time Applications,» 2003.

[14] IETF, «IAX: Inter-Asterisk eXchange Version 2,» 2010.

[15] IETF, «Media Gateway Control Protocol (MGCP),» 2003.

[16] International Telecommunication Union, *Recommendation ITU-T H.323,* 2009.

[17] H. &. M. P. Liu, «Voice over IP signaling: H.323 and beyond,» *Communications Magazine, IEEE,* 2000.

[18] International Telecommunication Union, «ITU-T Recommendation X.680 (07/2002),» 2002.

[19] International Telecommunication Union, «Recommendation X.691 (11/08),» 2008.

[20] International Telecommunication Union, *Recommendation ITU-T H.225.0,* 2009.

[21] International Telecommunication Union, *Recommendation ITU-T H.245,* 2011.

[22] Gary A. Thom, Dela Information Systems, Inc., «H323: The Multimedia

Communications for Local Area Networks,» *IEEE Communications Magazine,* nº Diciembre 1996, p. 5, 1996.

[23] B. Ward, The book of VMware: the complete guide to VMware workstation, San Francisco, 2002.

[24] P. &. C. E. Montoro, «A comparative study of VoIP standards with Asterisk,» de *Digital Telecommunications. ICDT '09. Fourth International Conference*, 20-25 July 2009.

Anexo I - **RAS Messages**

Terminal and GK discovery messages

Message name	Message description	Message type
GRQ	Gatekeeper Request	Request
GCF	Gatekeeper Confirm	Response
GRJ	Gatekeeper Reject	Response

Terminal and GW registration messages

Message name	Message description	Message type
RRQ	Registration Request	Request
RCF	Registration Confirm	Response
RRJ	Registration Reject	Response

Terminal/GK unregistration messages

Message name	Message description	Message type
URQ	Unregistration Request	Request
UCF	Unregistration Confirm	Response
URJ	Unregistration Reject	Response

Terminal to GK admission messages

Message name	Message description	Message type
ARQ	Admission Request	Request
ACF	Admission Confirm	Response
ARJ	Admission Reject	Response

Location request messages

Message name	Message description	Message type
LRQ	Location Request	Request
LCF	Location Confirm	Response
LRJ	Location Reject	Response

Disengage messages

Message name	Message description	Message type
DRQ	Disengage Request	Request
DCF	Disengage Confirm	Response
DRJ	Disengage Reject	Response

Status request messages

Message name	Message description	Message type
IRQ	Info Request	Request
IRR	Info Request Response	Response
IACK	Info Acknowledgement	Response

Terminal to gatekeeper requests for changes in bandwidth

Message name	Message description	Message type
BRQ	Bandwidth Request	Request
BCF	Bandwidth Confirm	Response
BRJ	Bandwidth Reject	Response

GW resource availability messages

Message name	Message description	Message type
RAI	Resource Availability Indication	Request
RAC	Resource Availability Confirmation	Response